8-1974

The Learn System for Computer-Assisted Instruction

Garret Alan Vander Lugt
Western Michigan University

Follow this and additional works at: https://scholarworks.wmich.edu/dissertations
Part of the Education Commons

Recommended Citation
https://scholarworks.wmich.edu/dissertations/2909

This Dissertation-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Dissertations by an authorized administrator of ScholarWorks at WMU. For more information, please contact maira.bundza@wmich.edu.
THE LEARN SYSTEM FOR
COMPUTER-ASSISTED INSTRUCTION

by

Garret Alan Vander Lugt

A Dissertation
Submitted to the
Faculty of The Graduate College
in partial fulfillment
of the
Degree of Doctor of Philosophy

Western Michigan University
Kalamazoo, Michigan
August 1974
FOREWORD

This dissertation contains two separate documents - the Learn Lesson Author's Guide and a listing of the programs that comprise the Learn system for computer-assisted instruction. Since it is intended that in the future these documents will be published separately, each has its own Table of Contents and page numbering system. In order to maintain the integrity of the two documents, the author has decided, with the consent of his Doctoral Advisory Committee, to depart from the standard dissertation format that has been established by The Graduate College at Western Michigan University. The overall Table of Contents for the dissertation has been eliminated, and single spacing has been used in the Guide in order to reduce the cost of printing it. The Acknowledgements have been included in the Guide since it will be more widely circulated than the program listing.
ACKNOWLEDGEMENTS

The author expresses his sincere appreciation to all those whose interest, encouragement, criticism, and hard work contributed to the development of the Learn system. In particular, the author expresses his gratitude to:

The members of his Doctoral Committee, Dr. Kenneth E. Dickie, Dr. John E. Herman, Dr. Paul E. Holkeboer, Dr. George G. Mallinson (chairman), and Professor Jack R. Meagher. (Special thanks go to Dr. John E. Herman, Dr. George G. Mallinson and Professor Jack R. Meagher for their constructive criticism during the writing of this document).

Mr. Eric Warren and Mr. Darwin Parrish for many hours of hard work spent in the design and construction of the teletype-slide projector interface.

Dr. Arthur Snapper, Dr. Stanley West, and especially Mr. Joe Liu for using the Learn system during its developmental stages.

Mrs. Janet Springer for excellent work in typing the manuscript, and Mrs. Karen Bushouse for all the time spent making Xerox copies of the manuscript and the Learn program listings.

His parents, who impressed upon him the importance of education and who encouraged him to go on to graduate school.

Finally, the author expresses his deepest appreciation to his wife Marie. Her understanding, encouragement, and love have not only contributed to the completion of the Learn project, but she has made the author's entire life richer and more meaningful.

Garret Alan Vander Lugt
PREFACE

This document is part of the author's dissertation that was submitted to the faculty of The Graduate College at Western Michigan University in partial fulfillment of the degree of Doctor of Philosophy. Also included in the dissertation is a cross-reference listing of all the programs that comprise the Learn system for computer-assisted instruction. A copy of the dissertation can be obtained from University Microfilms in Ann Arbor, Michigan.

The Learn system was developed at Western Michigan University during the period 1971-1974, and it has undergone extensive revision during this time. The current version, version 3, is described in this document. This version of Learn has been designed to run on a PDP-10 computer under the control of a 5.06 monitor, and it may be assembled for use with either a KA10 or K110 central processing unit. A teletype-slide projector interface was also developed as part of this project.

This document is intended for use by instructors who wish to prepare lessons for the Learn system and who may have only limited experience in programming computers. Chapter 1 describes the basic features of the Learn system, and Chapter 2 outlines the procedure that instructors must follow in order to use Learn effectively. Chapter 3 describes all of the Learn system commands in detail, and the commands that are used to edit Learn lessons are described in Chapter 4. Chapter 5 describes the Learn Instructional Language (LIL), which is somewhat similar to the PLANIT programming language. Chapters six through eight provide detailed information on Learn record keeping, lesson testing capabilities, and calculation mode usage, respectively.

Although Learn has been tested in a real educational environment by several instructors at Western Michigan University, no doubt there are a number of "bugs" that have gone undetected. Furthermore, since this is the first edition of the Learn Lesson Author's Guide, some explanations may not be completely clear, and some important information may have been omitted accidentally. Also time did not permit the inclusion of a number of desirable features in the Learn system such as string manipulation capabilities and more sophisticated input-output capabilities. The author hopes that these shortcomings will not be a source of undue frustration for those who attempt to use Learn.

G. V. L.

Kalamazoo
August, 1974
INFORMATION TO USERS

This material was produced from a microfilm copy of the original document. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the original submitted.

The following explanation of techniques is provided to help you understand markings or patterns which may appear on this reproduction.

1. The sign or "target" for pages apparently lacking from the document photographed is "Missing Page(s)". If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to insure you complete continuity.

2. When an image on the film is obliterated with a large round black mark, it is an indication that the photographer suspected that the copy may have moved during exposure and thus cause a blurred image. You will find a good image of the page in the adjacent frame.

3. When a map, drawing or chart, etc., was part of the material being photographed the photographer followed a definite method in "sectioning" the material. It is customary to begin photoing at the upper left hand corner of a large sheet and to continue photoing from left to right in equal sections with a small overlap. If necessary, sectioning is continued again — beginning below the first row and continuing on until complete.

4. The majority of users indicate that the textual content is of greatest value, however, a somewhat higher quality reproduction could be made from "photographs" if essential to the understanding of the dissertation. Silver prints of "photographs" may be ordered at additional charge by writing the Order Department, giving the catalog number, title, author and specific pages you wish reproduced.

5. PLEASE NOTE: Some pages may have indistinct print. Filmed as received.

Xerox University Microfilms
300 North Zeeb Road
Ann Arbor, Michigan 48106

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
VANDER LUGT, Garret Alan, 1945-
THE LEARN SYSTEM FOR COMPUTER-ASSISTED INSTRUCTION.

Western Michigan University, Ph.D., 1974
Education, theory and practice

Xerox University Microfilms, Ann Arbor, Michigan 48106

THIS DISSERTATION HAS BEEN MICROFILMED EXACTLY AS RECEIVED.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENTS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREFACE</td>
<td>i</td>
</tr>
<tr>
<td>CHAPTER 1 INTRODUCTION TO LEARN</td>
<td></td>
</tr>
<tr>
<td>1.1 Timesharing</td>
<td>1-1</td>
</tr>
<tr>
<td>1.2 Peripheral Devices</td>
<td>1-2</td>
</tr>
<tr>
<td>1.3 Access to Learn</td>
<td>1-2</td>
</tr>
<tr>
<td>1.4 Programming Capabilities</td>
<td>1-3</td>
</tr>
<tr>
<td>1.5 The Calculation Mode</td>
<td>1-3</td>
</tr>
<tr>
<td>1.6 Record Keeping Capabilities</td>
<td>1-3</td>
</tr>
<tr>
<td>1.7 Test Mode Features</td>
<td>1-4</td>
</tr>
<tr>
<td>1.8 Student Use of Learn</td>
<td>1-4</td>
</tr>
<tr>
<td>1.9 Some Technical Information</td>
<td>1-4</td>
</tr>
<tr>
<td>CHAPTER 2 PREPARING AND USING LEARN LESSONS</td>
<td></td>
</tr>
<tr>
<td>2.1 Entering Lessons into the Computer System</td>
<td>2-1</td>
</tr>
<tr>
<td>2.2 Processing a Lesson</td>
<td>2-2</td>
</tr>
<tr>
<td>2.3 Setting Up Record Files</td>
<td>2-3</td>
</tr>
<tr>
<td>2.4 Using Learn Lessons</td>
<td>2-4</td>
</tr>
<tr>
<td>2.5 Generating Learn Reports</td>
<td>2-4</td>
</tr>
<tr>
<td>CHAPTER 3 LEARN SYSTEM COMMANDS</td>
<td></td>
</tr>
<tr>
<td>3.1 The HELP Command</td>
<td>3-1</td>
</tr>
<tr>
<td>3.2 Lesson Editing Commands</td>
<td>3-2</td>
</tr>
<tr>
<td>3.2.1 The WRITE Command</td>
<td>3-2</td>
</tr>
<tr>
<td>3.2.2 The EDIT Command</td>
<td>3-3</td>
</tr>
<tr>
<td>3.3 The PROCESS Command</td>
<td>3-5</td>
</tr>
<tr>
<td>3.4 Lesson Execution Commands</td>
<td>3-7</td>
</tr>
<tr>
<td>3.4.1 The GET Command</td>
<td>3-7</td>
</tr>
<tr>
<td>3.4.2 The TEST Command</td>
<td>3-8</td>
</tr>
<tr>
<td>3.5 File Maintenance Commands</td>
<td>3-9</td>
</tr>
<tr>
<td>3.5.1 The COPY Command</td>
<td>3-10</td>
</tr>
<tr>
<td>3.5.2 The DELETE Command</td>
<td>3-12</td>
</tr>
<tr>
<td>3.5.3 The FILES Command</td>
<td>3-16</td>
</tr>
<tr>
<td>3.5.4 The LIST Command</td>
<td>3-17</td>
</tr>
<tr>
<td>3.5.5 The RENAME Command</td>
<td>3-18</td>
</tr>
<tr>
<td>3.6 Record Keeping Commands</td>
<td>3-20</td>
</tr>
<tr>
<td>3.6.1 The CLEAR Command</td>
<td>3-20</td>
</tr>
<tr>
<td>3.6.2 The UPDATE Command</td>
<td>3-22</td>
</tr>
<tr>
<td>3.6.3 The REPORT Command</td>
<td>3-23</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
TABLE OF CONTENTS
(continued)

<table>
<thead>
<tr>
<th>CHAPTER 3</th>
<th>LEARN SYSTEM COMMANDS (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7.1</td>
<td>The MONITOR Command</td>
</tr>
<tr>
<td>3.7.2</td>
<td>The KJOB Command</td>
</tr>
<tr>
<td>3.7.3</td>
<td>The BYE Command</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 4</th>
<th>EDITING LEARN LESSONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Entering the Learn Lesson Editor</td>
</tr>
<tr>
<td>4.2</td>
<td>Line Numbers</td>
</tr>
<tr>
<td>4.3</td>
<td>Learn Edit Commands</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The A Command</td>
</tr>
<tr>
<td>4.3.2</td>
<td>The C Command</td>
</tr>
<tr>
<td>4.3.3</td>
<td>The D Command</td>
</tr>
<tr>
<td>4.3.4</td>
<td>The I Command</td>
</tr>
<tr>
<td>4.3.5</td>
<td>The K Command</td>
</tr>
<tr>
<td>4.3.6</td>
<td>The N Command</td>
</tr>
<tr>
<td>4.3.7</td>
<td>The R Command</td>
</tr>
<tr>
<td>4.3.8</td>
<td>The T Command</td>
</tr>
<tr>
<td>4.3.9</td>
<td>The X Command</td>
</tr>
<tr>
<td>4.4</td>
<td>Exiting from the Learn Lesson Editor</td>
</tr>
<tr>
<td>4.5</td>
<td>Editing Efficiency</td>
</tr>
<tr>
<td>4.6</td>
<td>An Example of Lesson Editing</td>
</tr>
<tr>
<td>4.7</td>
<td>Summary of Learn Edit Commands</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 5</th>
<th>THE <em>LEARN</em> INSTRUCTIONAL _LANGUAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>LIL Frames</td>
</tr>
<tr>
<td>5.2</td>
<td>Q-frames</td>
</tr>
<tr>
<td>5.2.1</td>
<td>The TEXT Section</td>
</tr>
<tr>
<td>5.2.1.1</td>
<td>The TEXT Statement</td>
</tr>
<tr>
<td>5.2.1.2</td>
<td>The SLIDE Statement</td>
</tr>
<tr>
<td>5.2.1.3</td>
<td>The PAGE Statement</td>
</tr>
<tr>
<td>5.2.1.4</td>
<td>TEXT Section Switches</td>
</tr>
<tr>
<td>5.2.1.4.1</td>
<td>The CALC Statement</td>
</tr>
<tr>
<td>5.2.1.4.2</td>
<td>The HINT Statement</td>
</tr>
<tr>
<td>5.2.1.4.3</td>
<td>The REVIEW Statement</td>
</tr>
<tr>
<td>5.2.1.4.4</td>
<td>The RECORD Statement</td>
</tr>
<tr>
<td>5.2.1.4.5</td>
<td>The STORE Statement</td>
</tr>
<tr>
<td>5.2.1.4.6</td>
<td>The BOTH Statement</td>
</tr>
<tr>
<td>5.2.2</td>
<td>The ANSWER Section</td>
</tr>
<tr>
<td>5.2.2.1</td>
<td>The ANSWER Statement</td>
</tr>
<tr>
<td>CHAPTER 5</td>
<td>THE LEARN INSTRUCTIONAL LANGUAGE (continued)</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>5.2.2.2</td>
<td>Anticipated Answers</td>
</tr>
<tr>
<td>5.2.2.3</td>
<td>Unanticipated Answers</td>
</tr>
<tr>
<td>5.2.2.4</td>
<td>Zero Labels</td>
</tr>
<tr>
<td>5.2.2.5</td>
<td>The MATCH Statement</td>
</tr>
<tr>
<td>5.2.2.6</td>
<td>The DELIM Statement</td>
</tr>
<tr>
<td>5.2.2.7</td>
<td>Answer Comparison Switches</td>
</tr>
<tr>
<td>5.2.2.7.1</td>
<td>The EXTRA Statement</td>
</tr>
<tr>
<td>5.2.2.7.2</td>
<td>The ORDER Statement</td>
</tr>
<tr>
<td>5.2.2.7.3</td>
<td>The PHONIC Statement</td>
</tr>
<tr>
<td>5.2.3</td>
<td>The ACTION Section</td>
</tr>
<tr>
<td>5.2.3.1</td>
<td>The ACTION Statement</td>
</tr>
<tr>
<td>5.2.3.2</td>
<td>The F: Statement</td>
</tr>
<tr>
<td>5.2.3.3</td>
<td>The A: Statement</td>
</tr>
<tr>
<td>5.2.3.4</td>
<td>The R: Statement</td>
</tr>
<tr>
<td>5.2.3.5</td>
<td>The Q: Statement</td>
</tr>
<tr>
<td>5.2.3.6</td>
<td>The B: Statement</td>
</tr>
<tr>
<td>5.2.3.7</td>
<td>The G: Statement</td>
</tr>
<tr>
<td>5.2.3.8</td>
<td>The C: Statement</td>
</tr>
<tr>
<td>5.2.3.9</td>
<td>The P: Statement</td>
</tr>
<tr>
<td>5.2.3.10</td>
<td>The S: Statement</td>
</tr>
<tr>
<td>5.2.3.11</td>
<td>The E: Statement</td>
</tr>
<tr>
<td>5.2.3.12</td>
<td>The L: Statement</td>
</tr>
<tr>
<td>5.2.3.13</td>
<td>Action Labels</td>
</tr>
<tr>
<td>5.2.3.13.1</td>
<td>The Time Up Label</td>
</tr>
<tr>
<td>5.2.3.13.2</td>
<td>Hint Labels</td>
</tr>
<tr>
<td>5.2.3.14</td>
<td>Unconditional Actions</td>
</tr>
<tr>
<td>5.2.4</td>
<td>The REM Statement</td>
</tr>
<tr>
<td>5.2.5</td>
<td>Example Q-frames</td>
</tr>
<tr>
<td>5.3</td>
<td>C-frames</td>
</tr>
<tr>
<td>5.3.1</td>
<td>The Format of C-frame Statements</td>
</tr>
<tr>
<td>5.3.1.1</td>
<td>Line Numbers</td>
</tr>
<tr>
<td>5.3.1.2</td>
<td>Iteration Fields</td>
</tr>
<tr>
<td>5.3.1.3</td>
<td>Statement Fields</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Mathematical Expressions</td>
</tr>
<tr>
<td>5.3.2.1</td>
<td>Constants</td>
</tr>
<tr>
<td>5.3.2.2</td>
<td>Character Constants</td>
</tr>
<tr>
<td>5.3.2.3</td>
<td>Regular Variables</td>
</tr>
<tr>
<td>5.3.2.4</td>
<td>Array Variables</td>
</tr>
<tr>
<td>5.3.2.5</td>
<td>Special Variables</td>
</tr>
<tr>
<td>5.3.2.6</td>
<td>Operators</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS  
(continued)  

<table>
<thead>
<tr>
<th>CHAPTER 5</th>
<th>THE LEARN INSTRUCTIONAL LANGUAGE (continued)</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3.2.7</td>
<td>Mathematical Functions</td>
<td>5-26</td>
</tr>
<tr>
<td>5.3.2.8</td>
<td>Special Functions</td>
<td>5-27</td>
</tr>
<tr>
<td>5.3.2.8.1</td>
<td>The SEEN Function</td>
<td>5-27</td>
</tr>
<tr>
<td>5.3.2.8.2</td>
<td>The RIGHT, WRONG, and NEUT Functions</td>
<td>5-27</td>
</tr>
<tr>
<td>5.3.2.8.3</td>
<td>The ATIME Function</td>
<td>5-27</td>
</tr>
<tr>
<td>5.3.2.8.4</td>
<td>The PATH Function</td>
<td>5-28</td>
</tr>
<tr>
<td>5.3.2.8.5</td>
<td>The TAG Function</td>
<td>5-28</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Variable Definition Statements</td>
<td>5-28</td>
</tr>
<tr>
<td>5.3.3.1</td>
<td>The SCALAR Statement</td>
<td>5-29</td>
</tr>
<tr>
<td>5.3.3.2</td>
<td>The ARRAY Statement</td>
<td>5-29</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Assignment Statements</td>
<td>5-29</td>
</tr>
<tr>
<td>5.3.5</td>
<td>Input Statements</td>
<td>5-30</td>
</tr>
<tr>
<td>5.3.5.1</td>
<td>The INPUT Statement</td>
<td>5-30</td>
</tr>
<tr>
<td>5.3.5.2</td>
<td>The FILL Statement</td>
<td>5-31</td>
</tr>
<tr>
<td>5.3.6</td>
<td>Output Statements</td>
<td>5-31</td>
</tr>
<tr>
<td>5.3.6.1</td>
<td>The TYPE Statement</td>
<td>5-31</td>
</tr>
<tr>
<td>5.3.6.2</td>
<td>The ATYPE Statement</td>
<td>5-33</td>
</tr>
<tr>
<td>5.3.7</td>
<td>Branching Statements</td>
<td>5-34</td>
</tr>
<tr>
<td>5.3.7.1</td>
<td>The GOTO Statement</td>
<td>5-34</td>
</tr>
<tr>
<td>5.3.8</td>
<td>The IF Statement</td>
<td>5-34</td>
</tr>
<tr>
<td>5.3.9</td>
<td>Calling External Lessons</td>
<td>5-35</td>
</tr>
<tr>
<td>5.3.9.1</td>
<td>The CALL Statement</td>
<td>5-35</td>
</tr>
<tr>
<td>5.3.9.2</td>
<td>The RETURN Statement</td>
<td>5-35</td>
</tr>
<tr>
<td>5.3.10</td>
<td>The RESERVE Statement</td>
<td>5-36</td>
</tr>
<tr>
<td>5.3.11</td>
<td>The REM Statement</td>
<td>5-36</td>
</tr>
<tr>
<td>5.3.12</td>
<td>Example C-frames</td>
<td>5-36</td>
</tr>
<tr>
<td>5.4</td>
<td>R-frames</td>
<td>5-38</td>
</tr>
<tr>
<td>5.5</td>
<td>B-frames</td>
<td>5-39</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 6</th>
<th>RECORD KEEPING AND REPORT GENERATION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Internal Record Keeping</td>
<td>6-1</td>
</tr>
<tr>
<td>6.2</td>
<td>The Transaction File</td>
<td>6-2</td>
</tr>
<tr>
<td>6.3</td>
<td>The Master Record File</td>
<td>6-2</td>
</tr>
<tr>
<td>6.4</td>
<td>Storage of Answer Strings</td>
<td>6-2</td>
</tr>
<tr>
<td>6.5</td>
<td>Learn Reports</td>
<td>6-3</td>
</tr>
<tr>
<td>6.5.1</td>
<td>Answer Reports</td>
<td>6-3</td>
</tr>
<tr>
<td>6.5.2</td>
<td>Reports Based Upon Master File Information</td>
<td>6-3</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
### TABLE OF CONTENTS
(continued)

<table>
<thead>
<tr>
<th>CHAPTER 7</th>
<th>TESTING LEARN LESSONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Entering Test Mode</td>
</tr>
<tr>
<td>7.2</td>
<td>Test Mode Commands</td>
</tr>
<tr>
<td>7.2.1</td>
<td>The :n Command</td>
</tr>
<tr>
<td>7.2.2</td>
<td>The :E Command</td>
</tr>
<tr>
<td>7.2.3</td>
<td>The :: Command</td>
</tr>
<tr>
<td>7.3</td>
<td>An Example</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER 8</th>
<th>THE CALCULATION MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Entering the Calculation Mode</td>
</tr>
<tr>
<td>8.2</td>
<td>Calculation Mode Statements</td>
</tr>
<tr>
<td>8.3</td>
<td>Some Restrictions</td>
</tr>
<tr>
<td>8.4</td>
<td>Some Example Calculations</td>
</tr>
</tbody>
</table>

| APPENDIX A | THE SLIDE PROJECTOR INTERFACE | A-1    |
| APPENDIX B | PHONETIC TRANSFORMATIONS     | B-1    |
| APPENDIX C | INFORMATION FILES            | C-1    |
| APPENDIX D | A SUMMARY OF FILENAMES, EXTENSIONS, AND PROTECTION CODES | D-1 |
| APPENDIX E | LEARN RECORD FILE FORMATS    | E-1    |
| APPENDIX F | DEVICE NUMBERS              | F-1    |
| APPENDIX G | CHARACTER CODES             | G-1    |
| INDEX      |                                  | INDEX-1|

vii

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
CHAPTER 1

INTRODUCTION TO LEARN

Learn is a complete system for computer-assisted instruction that was developed by the author of this Guide as part of his doctoral research at Western Michigan University. The Learn system was designed for use on a PDP-10 computer running under the control of a timesharing monitor. Using Learn, PDP-10 users with only a limited knowledge of computer programming can enter lessons, that are in programmed instruction format, into the computer system for later presentation to students via remote terminals. Student performance data are stored automatically for later examination by lesson authors.

NOTE: In the examples shown in this document, computer output is underlined, and user input is not underlined. ESC or altmode characters are represented by â‚¬, and carriage return characters are represented by \\.

1.1 Timesharing

The PDP-10 is a large, timesharing computer that is capable of simultaneously interacting with a large number of users located at remote terminals. Terminals are either wired directly into the computer, or communications may take place over ordinary telephone lines. The computer executes only one user's program at a time, but it switches from one user to another so rapidly that each user has the illusion that he has the computer all to himself.

The heart of the PDP-10 system is the timesharing monitor. The monitor is a program that is in the computer's memory at all times, and that controls the running of all user programs. Users gain access to programs that are stored in the computer system by typing commands to the monitor. A complete discussion of monitor commands can be found in the DEC System 10 Users Handbook.²

¹,² PDP and DEC are trademarks of Digital Equipment Corporation, Maynard, Massachusetts.
1.2 Peripheral Devices

Learn lessons are typed into the computer at teletypewriters or equivalent devices, such as CRT displays, and are presented to students via these same devices. Slide projectors may be interfaced with student terminals and run automatically by the computer (see Appendix A). Slide projectors are useful for presenting graphs, drawings, photographs, and any other information that is difficult to convey via a teletypewriter.

Learn lessons and record files are stored on the disk while they are being used. The disk is a random-access, mass-storage device that allows extremely rapid retrieval of the information that is stored on it. In order to conserve disk space, lessons that are not being used should be transferred to magnetic tape or DEC tape for long term storage.

1.3 Access to Learn

In order to make use of the PDP-10, users must first log-in at a terminal that is in communication with the computer. (See the DEC System 10 Users Handbook for a description of the log-in procedure.) During log-in the user identifies himself to the monitor by typing in the project-programmer numbers and password that have been assigned to him by the computer installation. The monitor checks to make sure that the user's project-programmer numbers and password match and that his allocated funds are not exhausted. If the user identifies himself properly and his funds are not exhausted, the monitor types the message of the day followed by a period that indicates that it is ready to accept a command from the user.

To run the Learn program, the user types the following monitor command.

_.R LEARN ↓ ...the period is typed by the monitor

This command instructs the monitor to load the Learn program into the computer's memory and to begin executing it.

Learn begins by typing a greeting followed by a request for a command as shown below.

GOOD MORNING! THIS IS LEARN (301A).
TYPE H FOR HELP.

ENTER COMMAND
At this point any of the commands described in Chapter 3 may be typed in by the user.

1.4 Programming Capabilities

Learn lessons must be programmed in LIL, the Learn Instructional Language. LIL enables persons with little or no previous programming experience to program lessons easily for use on the Learn system. By using LIL statements, questions can be printed on student terminals, and slide projectors can be controlled. Student responses can be evaluated in terms of whether or not they contain key words specified by the lesson author, or if the response is numeric, it can be tested to determine if it lies within specified limits around the correct answer. Branching in an instructional sequence can depend on a student's response to a particular question, his response time, or his responses to a series of questions. It is also possible to randomly generate numbers for a problem, have the computer solve the problem using these numbers, present the problem to the student, and compare the student's answer to the computer's answer.

1.5 The Calculation Mode

Learn has a calculation mode that is available to students while they are working on a lesson. The calculation mode enables students to use the computer as a powerful desk calculator in order to solve mathematical problems that are presented to them. Built-in functions that are available include the square root, cube root, trigonometric functions, inverse trigonometric functions, exponential, and natural logarithm. Arrays of data can be typed in and stored for repeated use throughout a lesson. The usefulness of the calculation mode is enhanced by its ability to perform iterative procedures. For example, the sum of the squares of an array of numbers can be found by typing in a single calculation mode statement. A description of the calculation mode capabilities is contained in Chapter 8.

1.6 Record Keeping Capabilities

Records of student responses, response times, total time on a lesson, and calculation mode usage are kept automatically. These records may be used for making decisions to branch students to various instructional sequences within a lesson. The information is also stored on the disk for later examination by instructors or lesson authors. Reports can be generated in several different formats including a summary of the information on each lesson, a summary of the information on each student, and detailed reports giving information on each question within a lesson. Chapter 6
describes the record keeping capabilities in detail.

1.7 Test Mode Features

Lesson authors are able to execute lessons in test mode in order to check them out before making them available to their students. While in test mode, lesson authors can alter the normal sequence of lesson execution. In this way previously tested frames can be skipped, and frames can be repeated any number of times to try out different possible student responses. See Chapter 7 for more information on the test mode.

1.8 Student Use of Learn

Students who use Learn need have no knowledge of computer programming or of the details of the Learn System, but they must be familiar with the following:

1) How to operate a terminal
2) How to log-in
3) How to run Learn by typing the R LEARN monitor command
4) How to execute a lesson by typing the GET command (see section 3.4.1)
5) How to log-out by typing the BYE command (see section 3.7.3)

Students will be more sophisticated users of Learn if they are also familiar with the following:

1) How to use the calculation mode (see Chapter B)
2) How to request a hint by typing a "?" (see section 5.2.1.4.2)
3) How to review portions of a lesson (see section 5.4)
4) How to stop in the middle of a lesson and reenter it at a later time (see section 5.5)

NOTE: Student use of Learn is simplified if their instructor's project number and programmer number are identical, and if all the students' project numbers are equal to their instructor's project number.

1.9 Some Technical Information

Learn system programs are written in MACRO-10, the assembly language of the PDP-10 computer. The coding is re-entrant so that any number of users can share one copy of a program in the computer's memory space. Each user has his own low segment for
storing individual data, but he shares with any number of other users a common high segment that contains only instructions and constants that are not altered during execution of the program. The Learn system is composed of four sharable high segments - the Lesson Editor, the Lesson Processor, the Lesson Interpreter, and the Report Generator. A sharable high segment, sometimes termed an overlay, is loaded into the computer's memory only if it is needed to execute a Learn command. For example, if several students are working on lessons and no other Learn functions are being performed, then only the Lesson Interpreter will be in core. The re-entrant feature results in a considerable saving of core space. For example, five students require only 17K of core (five 2K low segments and one sharable, 7K high segment) rather than the 45K which would be required if the programs were not re-entrant.

Before lessons can be interpreted by the Lesson Interpreter, they must be processed. The Lesson Processor checks each LIL statement in a lesson for proper format and transforms it into a form which can be interpreted more efficiently by the Lesson Interpreter. After processing two versions of the lesson exist - an unprocessed version and a processed version. Only the unprocessed lesson can be edited, and it must be reprocessed after any changes are made. The processed version is the version used by students. A single copy of the processed lesson can be stored on the disk and used simultaneously by any number of students.
CHAPTER 2

PREPARING AND USING LEARN LESSONS

This chapter describes the basic procedure that must be followed in order to prepare and use Learn lessons. No attempt is made to provide instruction in the preparation of programmed instruction materials. However, prior to entering the lessons into the computer, the same steps should be followed in preparing Learn lessons as are followed in preparing ordinary programmed lessons. The quality of a Learn lesson depends to a great extent on the amount of careful planning and writing that is done before the lesson is typed into the computer system. Readers who are not familiar with the preparation of programmed materials should consult one of the books that are available on this topic.1

2.1 Entering Lessons into the Computer System

After a lesson has been written on paper in the Learn Instructional Language, it must be typed into the computer system at a terminal. This is done by using the Learn WRITE command (see section 3.2.1). The WRITE command causes Learn to read in the lesson as it is typed by the user and to store it on the disk. A filename consisting of up to six characters is assigned to the lesson by the WRITE command. Learn also assigns a <177> protection code to the lesson so that it will not be deleted when the user logs off the computer.

Frames are numbered automatically, beginning with ten and incrementing by ten, as they are typed in by the user. Learn indicates that it is ready to accept a line of characters from the user's terminal by typing a line number followed by a > sign. In addition, Learn types the entire first line of each frame except for the last character, which indicates the frame type - Q, C, R, or B. A frame is terminated by typing the ESC (or altmode) key on the line following the last line in the frame. A lesson is terminated by typing the ESC (or altmode) key at the end of the first line instead of typing one of the letters Q, C, R, or B.

After the lesson has been terminated, Learn types a > sign. At this point, the user may type any of the edit commands which are described in Chapter 4, or he may type E to exit from the lesson editor. The example below shows how a lesson can be entered into the computer system.

...the user runs Learn

GOOD EVENING!  THIS IS LEARN (301A).
TYPE H FOR HELP.

ENTER COMMAND
*WRITE LESN1
...the user enters a lesson called LESN1

1 10 TYPE->Q
2 >TEXT
3 >WHO WAS THE FIRST PRESIDENT OF THE UNITED STATES?
4 
5 >ANSWER
6 >WASHINGTON
7 >ACTIONS
8 >A
9 >F:
10> @
11> F:
12> A:
13> $ ...user types ESC to end the frame

1 20 TYPE->Q
2 >TEXT
3 >WHO WAS PRESIDENT DURING THE CIVIL WAR?
4 
5 >ANSWER
6 >LINCOLN
7 >ACTIONS
8 >A
9 >F:
10>@
11> F:
12> A:
13> $ ...user types ESC to end the frame

1 30 TYPE->
> E ...user types ESC to end the lesson

ENTER COMMAND
* ...Learn is ready to accept a command

2.2 Processing a Lesson

After a lesson has been typed into the computer system, it must be processed before it can be used by students. Lessons are processed using the PROCESS command (see section 3.3). The example lesson shown above could be processed using the following command.

2-2
ENTER COMMAND
*PROCESS LESN1

TOTAL ERRORS DETECTED: 0

During processing Learn checks each LIL statement for proper syntax. If no errors are found, then the message "TOTAL ERRORS DETECTED: 0" is printed on the user's terminal. If errors are found, messages that include the frame number, the line number, and the error type are printed on the user's terminal. These errors can be corrected by using the edit commands that are described in Chapter 4.

After a lesson has been processed with no errors detected, it is ready for student use. A single copy of the processed lesson may be used simultaneously by any number of students. Normally the processed lesson is stored under an instructor's identical project-programmer numbers.

2.3 Setting Up Record Files

In order for student performance data to be stored, it is necessary for instructors to set up record files into which the data can be written. This is done by using the CLEAR command (see section 3.6.1) as shown below.

ENTER COMMAND
*CLEAR

ARE YOU SURE? YES

TRANSACTION FILE CLEARED

The form of the clear command shown above sets up a transaction file in which student performance data can be stored. These data include the number of correct and incorrect answers, answering times, and answer labels.

If the instructor also wishes to store the entire string of characters which a student types in responding to each frame in a lesson, then an answer file for that lesson must be created. Answer files are created using the CLEAR command with the lesson name following CLEAR as shown below.

ENTER COMMAND
*CLEAR LESN1

ARE YOU SURE? YES
ANSWER FILE CLEARED

In order to store answer strings, the lesson author must also include the statement STORE ON> (see section 5.2.1.4.5) in the first Q-frame in the lesson.

2.4 Using Learn Lessons

After a lesson has been processed and record files have been established, students may begin using the lesson. Students gain access to a lesson by typing the GET command (see section 3.4.1). An example GET command is shown below. In this example, it is assumed that the lesson is stored under the instructor's identical project-programmer numbers, and that the student's project number is the same as the instructor's project number.

```
*R LEARN ↓
GOOD MORNING/ THIS IS LEARN (301A).
TYPE H FOR HELP.

ENTER COMMAND
*GET LESN1 ↓

WHO WAS THE FIRST PRESIDENT OF THE UNITED STATES?
*WASHINGTON ↓

FINE!

WHO WAS PRESIDENT DURING THE CIVIL WAR?
*GRANT ↓

YOU GOOFED!
ANS: LINCOLN

ENTER COMMAND
*BYE ↓
```

2.5 Generating Learn Reports

Instructors may obtain reports based upon student performance data at any time. The first step in obtaining a report is to type in an UPDATE command (see section 3.6.2). An UPDATE causes the information in the transaction file to be transferred to the master record file. If no master file exists when the UPDATE is issued, one is created. An example is shown below.

2-4
ENTER COMMAND
~UPDATE ~

MASTER FILE UPDATED

After the master file has been updated, reports can be generated by using the REPORT command (see section 3.6.3). For example, to obtain a summary report on a terminal the following command could be used.

ENTER COMMAND
*REPORT TTY:LESN1 ~

The information in the report would include for each student total elapsed time for the lesson (E. TIME), total answering time (A. TIME), the number of correct answers, and the number of incorrect answers.

A list of the answers that were typed in by each student can be obtained by using the REPORT command followed by the /A switch. An example is shown below.

ENTER COMMAND
*REPORT TTY:LESN1/A ~

See Chapter 6 for a more complete description of the REPORT command.
CHAPTER 3

LEARN SYSTEM COMMANDS

The various commands that are available to Learn users are described in this chapter. Learn indicates that it is ready to accept a command by typing the following message on the user's terminal.

ENTER COMMAND

Commands must be terminated by a carriage return. Any number of extra spaces (not tabs) may be included in a command string, and the first word in a command string may be abbreviated provided that enough letters are typed to distinguish it from other commands. For example, WRITE may be abbreviated by typing W. If not enough letters are typed, the message "AMBIGUOUS ABBREVIATION" is printed on the user's terminal. If the first word in a command is not recognized by Learn, the message "ILLEGAL COMMAND. TRY AGAIN." is printed.

NOTE: Users cannot exit from Learn to monitor level by typing control C. The MONITOR command (see section 3.7.1) should be used for this purpose.

3.1 The HELP Command

Function

The HELP command is used to execute the HELP lesson which contains basic information about the Learn system.

Format

HELP

There are no arguments for this command.

Characteristics

The processed HELP lesson (HELP.PRS) is stored on device SYS along with the four re-entrant high segments that make up the Learn system.

Learn does not store student performance data at the conclusion of the HELP lesson.

3-1
3.2 Lesson Editing Commands

There are two commands that can be used to gain access to the Learn Lesson Editor. The WRITE command is used to enter a new lesson into the computer system, and the EDIT command is used to make changes in a previously entered lesson.

3.2.1 The WRITE Command

**Function**

The WRITE command is used to enter a new lesson into the computer system.

**Format**

```
WRITE lesson /inc
```

lesson - the filename that the lesson author wishes to assign to the lesson

inc - an optional argument which determines the frame number increment

**Characteristics**

The WRITE command transfers control to the Learn Lesson Editor that stores on the disk all lesson statements typed by the author.

A <177> protection code is assigned to the lesson file.

The lesson filename may not contain more than six characters, and it may not begin with a numeral.

If inc is omitted, the Lesson Editor automatically assigns frame numbers beginning with ten and incrementing by ten. If inc is included, the Lesson Editor assigns frame numbers beginning with inc and incrementing by inc.

**Messages**

LESSON NAMES MAY NOT BEGIN WITH A NUMERAL
The lesson filename specified by the user begins with one of the characters 0-9.

TOO MANY CHARACTERS IN LESSON NAME

The lesson filename specified by the user contains more than six characters.

NO LESSON NAME WAS GIVEN

No lesson filename follows WRITE.

LESSON WITH THAT NAME ALREADY EXISTS

The filename of the new lesson is the same as the filename of an already existing unprocessed lesson that is stored under the user's project-programmer numbers.

ILLEGAL FRAME INCREMENT

The frame increment is not a positive integer or is not in proper format.

Examples

ENTER COMMAND
*WRITE PHYS1

1 10 TYPE-> ...Lesson Editor numbers 1st frame 10

* * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*WRITE CHEM/5

1 5 TYPE-> ...Lesson Editor numbers 1st frame 5

3.2.2 The EDIT Command

Function

The EDIT command is used to make alterations in unprocessed lesson files.

Format

EDIT lesson /inc

lesson - the filename which was assigned to the lesson
inc - an optional argument which determines the frame number increment

Characteristics

The EDIT command transfers control to the Learn Lesson Editor that executes any edit commands typed by the user.

The frame increment applies only to frames which are appended to the lesson. It need not be the same as the increment of already existing frames. If inc is omitted, the Lesson Editor increments the frame numbers of appended frames by ten.

Only lessons stored under the user's own project-programmer numbers may be edited.

Messages

LESSON NAMES MAY NOT BEGIN WITH A NUMERAL

The lesson filename specified by the user begins with one of the characters 0-9.

TOO MANY CHARACTERS IN LESSON NAME

The lesson filename specified by the user contains more than six characters.

LESSON NOT FOUND.

The specified lesson cannot be found under the user's project-programmer numbers.

ILLEGAL FRAME INCREMENT

The frame increment is not a positive integer or is not in proper format.

Examples

ENTER COMMAND
*EDIT BIOL2

> ...Learn is ready to accept an edit command

For this example, assume that the last frame of the lesson before editing is numbered 80.
ENTER COMMAND *EDIT GEOL1/5

A 85 TYPE->

...user types the A command to append frames onto the lesson

...Learn numbers the appended frame 85

3.3 The PROCESS Command

Function

The PROCESS command is used to process all or part of an unprocessed lesson. Lessons must be processed before they can be used by students.

Format

PROCESS lesson \( \{ /n \} \)

\( /n-m \)

lesson - the filename of the lesson that the user wishes to process

n - an optional argument that specifies the frame number of the first frame to be processed

m - an optional argument that specifies the frame number of the last frame to be processed

Characteristics

The PROCESS command transfers control to the Learn Lesson Processor that processes the unprocessed lesson file.

Only unprocessed lesson files that are stored under the user's own project-programmer numbers may be processed.

A <155> protection code is assigned to the processed lesson file.

The filename of the processed lesson is the same as the filename of the unprocessed lesson, and the extension is PRS.

If neither m nor n is included in a PROCESS command, all the frames are processed. If only n is included, only frame n is processed.

An unprocessed lesson file is not processed if a processed lesson file with the same name has a later creation time than the unprocessed file. The creation time of an unprocessed lesson file is updated each time it is edited.

3-5
Messages

NO LESSON NAME WAS GIVEN

No lesson name follows PROCESS.

MORE THAN SIX CHARACTERS IN LESSON NAME

The lesson filename specified by the user contains more than six characters.

ILLEGAL FRAME NUMBER IN PROCESS SWITCH

Either n or m is not a positive integer or is not in proper format.

LESSON NOT FOUND

The unprocessed lesson with the specified name cannot be found under the user's project-programmer numbers.

NO FRAMES WERE PROCESSED

There is no frame in the lesson which has a frame number in the range n-m.

TOTAL ERRORS DETECTED: 0

No errors were detected during lesson processing. If errors are detected, appropriate error messages are typed on the user's terminal.

Examples

ENTER COMMAND
*PROCESS BIOL1

TOTAL ERRORS DETECTED: 0

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*PROCESS CHEM/100-250

...only frames 100-250 are processed

FRAME 200, LINE 21
BRANCH TO NONEXISTENT FRAME - 30

TOTAL ERRORS DETECTED: 1
3.4 Lesson Execution Commands

There are two commands that can be used to execute a lesson. The GET command is used by students to execute a lesson that has been stored on the disk by their instructor, and the TEST command is used by lesson authors to execute a lesson in test mode (see Chapter 7).

3.4.1 The GET Command

Function

Students use the GET command to execute lessons.

Format

GET lesson \[p,q\]

lesson - the filename of the lesson that the student wishes to execute

p - the project number under which the lesson is stored

q - the programmer number under which the lesson is stored

Characteristics

If the project-programmer numbers are omitted, Learn searches for the lesson under identical project and programmer numbers, both of which are equal to the student's project number. Normally an instructor should have identical project-programmer numbers, and his students' project numbers should be the same as his project number. If this is the case, students can omit \[p,q\] when executing a lesson that is stored under their instructor's project-programmer numbers.

Users who do not have identical project-programmer numbers must include their own project-programmer numbers in the GET command in order to execute a lesson that is stored under their own project-programmer numbers.

The GET command transfers control to the Learn Lesson Interpreter that interprets the processed lesson file. Only processed lesson files can be executed.

Messages

NO LESSON NAME WAS GIVEN

No lesson filename follows GET.
TOO MANY CHARACTERS IN LESSON NAME

More than six characters are included in the lesson filename.

ILLEGAL PROJ-PROG NUMBERS

The project-programmer numbers are not in proper format.

LESSON NOT FOUND

The processed lesson file cannot be found under the specified
project-programmer numbers.

Examples

In the examples below, assume that the user's project-programmer
numbers are 575,600.

ENTER COMMAND
*GET GEOL5
...Learn searches for the lesson
under 575,575

THIS LESSON DEALS WITH...
...lesson execution begins

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*GET BIOL [575,600]
...Learn searches for the lesson
under user's own proj-prog
numbers

THE THEORY OF...

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*GET ATOM [360,425]
...Learn searches for the lesson
under 360,425

3.4.2 The TEST Command

Function

The TEST command is used by lesson authors to execute lessons in
test mode.

Format

\[ \text{TEST lesson} \{ /n \} \]
\[ \{ /n-m \} \]
lesson - the filename of the lesson that the user wishes to test

n - an optional argument that specifies the frame number of the first frame to be processed

m - an optional argument that specifies the frame number of the last frame to be processed

**Characteristics**

The TEST command transfers control to the Learn Lesson Processor that processes the lesson. After processing, control is transferred to the Lesson Interpreter that executes the lesson in test mode (see Chapter 7 for a description of test mode capabilities).

If errors are detected during lesson processing, lesson execution is aborted.

The TEST command has the same characteristics as the PROCESS command (see section 3.3).

**Messages**

TEST command messages are the same as PROCESS command messages (see section 3.3).

**Examples**

```
ENTER COMMAND
*TEST ACIDS

TOTAL ERRORS DETECTED: 0

START AT FRAME *80
...user elects to start at frame 80

* * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*TEST BASES/300-400
...only frames 300-400 will be processed

TOTAL ERRORS DETECTED: 0

START AT FRAME *300
```

3.5 File Maintenance Commands

There are five commands which aid users in maintaining Learn files. These commands are COPY, DELETE, FILES, LIST, and RENAME.
3.5.1 The COPY Command

Function

The COPY command is used to make a copy of a Learn file.

Format

\[
\text{COPY file2=file1} \begin{cases} \\
\text{ /U} \\
\text{ /P} \\
\text{ /R} \\
\text{ /A} \\
\end{cases}
\]

file1 - the filename of the file being copied

file2 - the filename of the new file

U, P, R, and A are optional switches that indicate the kind of file being copied.

Characteristics

Only files that are stored on the disk under the user's own project-programmer numbers can be copied.

The meanings of the optional switches are as follows:

- U - copy an unprocessed lesson file
- P - copy a processed lesson file
- R - copy a record file
- A - copy an answer file
- no switch - copy an unprocessed lesson file

Up to four switches may be included in a single copy command.

Messages

ILLEGAL COPY COMMAND

The COPY command is not in proper format.

NO FILENAME WAS GIVEN

One of the two filenames is missing.

FILENAMES MAY NOT BEGIN WITH A NUMERAL

The new filename begins with one of the characters 0-9.

ILLEGAL SWITCH
A switch other than U, P, R, or A is included in the COPY command.

**ILLEGAL FILENAME**

One of the filenames contains more than six characters, or the first filename is not followed by an equals sign.

**UNPROCESSED LESSON NOT FOUND**

The specified lesson file cannot be found under the user's project-programmer numbers.

**PROCESSED LESSON NOT FOUND**

The processed lesson cannot be found under the user's project-programmer numbers.

**RECORD FILE NOT FOUND**

The specified record file cannot be found under the user's project-programmer numbers.

**ANSWER FILE NOT FOUND**

The specified answer file cannot be found under the user's project-programmer numbers.

**UNPROCESSED LESSON COPIED**

Copying of the unprocessed lesson file is complete.

**PROCESSED LESSON COPIED**

Copying of the processed lesson file is complete.

**RECORD FILE COPIED**

Copying of the record file is complete.

**ANSWER FILE COPIED**

Copying of the answer file is complete.

**Examples**

**ENTER COMMAND**

*COPY TEMP=PHYS2↓

**UNPROCESSED LESSON COPIED**

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
3.5.2 The DELETE Command

The DELETE command is used to erase Learn files from the disk. It is also used to delete selected information from the master record file (see section 6.3).

Format

1) DELETE filename
   \(\{/U\} \{/P\} \{/B\} \{/R\} \{/A\}\)

2) DELETE lesson
   \(\{/p\} \{(ml/dl/yl)\} \{/p-q\} \{(ml/dl/yl-m2/d2/y2)\} \{/[f1]\} \{/[f1-f2]\}\) /I

filename - name of the file to be deleted

U, P, B, R, and A are optional switches that indicate which kind of file is to be deleted.

lesson - filename of a lesson (optional)

p and q - student programmer numbers (optional)

ml/dl/yl - date in the form month/day/year (optional)

m2/d2/y2 - date in the form month/day/year (optional)

f1 and f2 - lesson frame numbers (optional)

I - required switch for deleting information

Characteristics

The first form of the DELETE command is used to delete entire files from the disk. The second form is used to delete selected information from the master record file (MASTER.REC).

Only files stored under the user's own project-programmer numbers may be deleted.

Before a DELETE command is executed, the message "ARE YOU SURE?" is printed on the user's terminal. If the user responds by typing YES (or any word beginning with Y), the deletion is completed;
otherwise, the message "NO INFORMATION DELETED" is printed on the user's terminal.

The meanings of the optional switches are as follows:

- U - delete an unprocessed lesson file
- P - delete a processed lesson file
- B - delete both a processed and unprocessed lesson file
- R - delete a record file
- A - delete an answer file
- no switch - delete an unprocessed file

Each optional argument that is included in the second form of the DELETE command restricts the scope of the command. For example, if a lesson filename is specified, performance data will be deleted from the master file only if they were produced by a student working on the specified lesson. If the lesson filename is omitted, information will be deleted without regard to the lesson to which it pertains.

The optional date refers to the date on which lesson execution began.

Messages

- TOO MANY CHARACTERS IN LESSON NAME
  The lesson filename contains more than six characters.
  
  ILLEGAL COMMAND. TRY AGAIN.

- The DELETE command is not in proper format.
  
  ILLEGAL STUDENT NUMBER
  A student programmer number is not a proper octal number or is not in proper format.
  
  ILLEGAL DATE FORMAT
  One of the specified dates is not in proper format.
  
  ILLEGAL FRAME NUMBER FORMAT
  The frame numbers are not in proper format.
  
  ILLEGAL SWITCH
  The switch is not one of the letters U, P, B, R, or A, or the switch is not in proper format.
INFORMATION DELETED
The deletion of the specified information is complete.

NO INFORMATION DELETED.
The specified information did not exist in the master record file.

UNPROCESSED LESSON NOT FOUND
The unprocessed lesson file cannot be found under the user's project-programmer numbers.

PROCESSED LESSON NOT FOUND
The processed lesson file cannot be found under the user's project-programmer numbers.

RECORD FILE NOT FOUND
The specified record file cannot be found under the user's project-programmer numbers.

ANSWER FILE NOT FOUND
The specified answer file cannot be found under the user's project-programmer numbers.

TRANSACTION FILE NOT FOUND
The transaction file (TRANS.REC) cannot be found under the user's project-programmer numbers.

MASTER FILE NOT FOUND
The master record file (MASTER.REC) cannot be found under the user's project-programmer numbers.

UNPROCESSED LESSON DELETED
Deletion of the unprocessed lesson file is complete.

PROCESSED LESSON DELETED
Deletion of the processed lesson file is complete.

RECORD FILE DELETED
Deletion of the record file is complete.
ANSWER FILE DELETED

Deletion of the answer file is complete.

TRANSACTION FILE DELETED

Deletion of the transaction file is complete.

MASTER FILE DELETED

Deletion of the master record file is complete.

Examples

ENTER COMMAND
*DELETE LESN1

ARE YOU SURE? YES

UNPROCESSED LESSON DELETED

ENTER COMMAND
*DELETE MASTER/R

ARE YOU SURE? YES

MASTER FILE DELETED

* * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*DELETE LESN2,24000-24040/I

ARE YOU SURE? YES

INFORMATION DELETED

* * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*DELETE 6000-6030(6/12/74-6/20/74)/I

ARE YOU SURE? YES

INFORMATION DELETED

* * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*DELETE LESN5 [50-100]/I

3-15
ARE YOU SURE? YES

INFORMATION DELETED

3.5.3 The FILES Command

Function

The FILES command is used to obtain a list of Learn files that are stored in the user's disk area.

Format

\[
\text{FILES} \begin{cases} 
/U \\ 
P \\ 
/R \\ 
/A \\
\end{cases}
\]

U, P, R, and A are optional switches that control the kind of files to be listed.

Characteristics

Three columns of information are printed on the user's terminal. The first column contains the filenames, the second column contains the file sizes in disk blocks, and the third column contains the creation dates.

The meanings of the optional switches are as follows:

- U - list unprocessed lesson files
- P - list processed lesson files
- R - list record files
- A - list answer files
- no switch - list all files

Up to four switches may be included in a single FILES command.

Messages

ILLEGAL SWITCH

A switch consists of a letter other than U, P, R, or A, or the switch is not in proper format.

NO FILES FOUND

There are no files of any kind stored under the user's project-programmer numbers.
Example

ENTER COMMAND
*FILES/UP

UNPROCESSED LESSONS

<table>
<thead>
<tr>
<th>Lesson</th>
<th>Copies</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>LESN1</td>
<td>20</td>
<td>6/10/74</td>
</tr>
<tr>
<td>LESN2</td>
<td>25</td>
<td>6/14/74</td>
</tr>
</tbody>
</table>

PROCESSED LESSONS

<table>
<thead>
<tr>
<th>Lesson</th>
<th>Copies</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>LESN1</td>
<td>19</td>
<td>6/11/74</td>
</tr>
<tr>
<td>LESN2</td>
<td>23</td>
<td>6/14/74</td>
</tr>
</tbody>
</table>

3.5.4 The LIST Command

Function

The LIST command is used to obtain printouts of Learn lessons on the line printer.

Format

LIST lesson /n

lesson - the filename of the lesson to be listed
n - an optional argument that specifies the number of copies

Characteristics

The unprocessed lesson file must be in the user's disk area in order to obtain a listing.

Learn produces a list file suitable for output on the line printer. This file is placed in the line printer output queue and is deleted after printing is complete.

If n is omitted, only one copy of the listing will be printed. A maximum of 63 copies will be printed.

Messages

NO FILENAME WAS GIVEN

No filename follows LIST.

ILLEGAL FILENAME

3-17
The filename of the lesson contains more than six characters.

ILLEGAL SWITCH

The switch was not a positive integer or was not in proper format.

LESSON NOT FOUND

The unprocessed lesson file cannot be found under the user's project-programmer numbers.

Examples

ENTER COMMAND
*LIST LESN1 J

LIST FILE IN PRINT QUEUE / COPIES: 1

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*LIST LESN2/3 J

LIST FILE IN PRINT QUEUE / COPIES: 3

3.5.5 The RENAME Command

The RENAME command is used to change the filename of a Learn file.

Format

```
RENAME name2=namel \n/U
/P
/R
/A
```

name1 - old filename

name2 - new filename

U, P, R, and A are optional switches that indicate the kind of file to be renamed.

Characteristics

Only files that are stored under the user's own project-programmer numbers may be renamed.

Filenames may contain a maximum of six characters, and they may

3-18
not begin with a numeral.

The meanings of the optional switches are as follows:

- U - rename an unprocessed lesson file
- P - rename a processed lesson file
- R - rename a record file
- A - rename an answer file
- no switch - rename an unprocessed lesson file

Up to four switches may be included in a single RENAME command.

Messages

NO FILENAME WAS GIVEN
One of the two filenames is missing from the command.

ILLEGAL FILENAME
One of the filenames contains more than six characters, or the first filename is not followed by an equals sign.

FILENAMES MAY NOT BEGIN WITH A NUMERAL
A filename begins with one of the characters 0-9.

UNPROCESSED LESSON NOT FOUND
The unprocessed lesson file cannot be found in the user's disk area.

PROCESSED LESSON NOT FOUND
The processed lesson file cannot be found in the user's disk area.

RECORD FILE NOT FOUND
The record file cannot be found in the user's disk area.

ANSWER FILE NOT FOUND
The answer file cannot be found in the user's disk area.

UNPROCESSED LESSON RENAMED
Renaming of the unprocessed lesson file is complete.

PROCESSED LESSON RENAMED

3-19
Renaming of the processed lesson file is complete.

RECORD FILE RENAMED

Renaming of the record file is complete.

ANSWER FILE RENAMED

Renaming of the answer file is complete.

Examples

ENTER COMMAND
*RENAME ROCKS=LESN1

UNPROCESSED LESSON RENAMED

* * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*RENAME PHYS2=LESN2/UP

UNPROCESSED LESSON RENAMED

PROCESSED LESSON RENAMED

3.6 Record Keeping Commands

The three commands that are used for setting up record files and generating reports on student performance are CLEAR, UPDATE, and REPORT.

3.6.1 The CLEAR Command

Function

The CLEAR command is used to establish a transaction file into which student performance data can be written, or to establish answer files into which student answer strings can be written.

Format

CLEAR lesson

lesson - filename of the lesson for which answers are to be stored (optional)
Characteristics

If the filename is omitted, the CLEAR command applies to the transaction file (TRANS.REC). All information in the transaction file is deleted, but the file itself is retained on the disk. If no transaction file exists when the CLEAR command is issued, one is created. Learn is able to store student performance data only if a transaction file has been created.

If the lesson filename is included in the CLEAR command, the answer file for the specified lesson is cleared. All information in the answer file is deleted, but the file itself is retained on the disk. If the answer file does not exist, one is created. Learn is able to store answers for a lesson only if an answer file for that lesson has been created.

The transaction file and answer files are assigned append only (A447) protection.

Messages

TOO MANY CHARACTERS IN LESSON NAME
There are more than six characters in the lesson filename.

TRANSACTION FILE CLEARED
Clearing of the transaction file is complete.

ANSWER FILE CLEARED
Clearing of the answer file is complete.

NO ROOM ON DISK
The disk is temporarily full. No files can be created.

Examples

ENTER COMMAND *CLEAR J
...user clears the transaction file

TRANSACTION FILE CLEARED

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND *CLEAR LESN3 J
...user clears the answer file for LESN3

3-21
3.6.2 The UPDATE Command

Function

The UPDATE command is used to append data from the transaction file onto the end of the master record file (MASTER.REC).

Format

    UPDATE

There are no arguments for this command.

Characteristics

Both the transaction file and the master record file must be stored under the user's own project-programmer numbers.

If no master file exists, one is created.

The UPDATE command clears the transaction file after the information has been transferred to the master file.

Learn reports will contain only information which has been transferred to the master record file by an UPDATE command.

Messages

    TRANSACTION FILE NOT FOUND

The transaction file cannot be found under the user's project-programmer numbers.

    MASTER FILE UPDATED

Updating of the master file is complete.

Example

ENTER COMMAND
*UPDATE

MASTER FILE UPDATED
3.6.3 The REPORT Command

Function

The REPORT command is used to obtain Learn reports on a terminal or on the line printer.

Format

\[
\text{REPORT } \begin{cases}
\text{TTY:} & \text{lesson } \{ p \} \{ (m_1/d_1/y_1) \} \{ f_1 \} \{ /A \} \\
\text{TTL:} & \{ p-q \} \{ (m_1/d_1/y_1-m_2/d_2/y_2) \} \{ f_1-f_2 \} \{ /S \} \\
\text{LPT:} & \{ /L \}
\end{cases}
\]

TTY: - output summary report on user's terminal (optional)
TTL: - output summary report on line printer (optional)
LPT: - output detailed report on line printer (optional)

lesson - lesson detailed report on line printer (optional)
p and q - student programmer numbers (optional)

m_1/d_1/y_1 - date in the form month/day/year (optional)
m_2/d_2/y_2 - date in the form month/day/year (optional)
f_1 and f_2 - lesson frame numbers (optional)

A, S, and L are optional switches that indicate the kind of report to be generated.

Characteristics

The master record file and answer files must be stored under the user's own project-programmer numbers in order to generate reports.

Each of the optional arguments that is included in a REPORT command restricts the scope of the information that is included in the report.

Although the lesson filename and the student programmer numbers are optional, at least one of these two arguments must be included in a REPORT command.

If none of the three arguments, TTY, TTL, or LPT, is included in the command, TTY is assumed.

The meanings of the optional switches are as follows:

3-23
A - generate an answer report
S - generate a student report
L - generate a lesson report

If the optional switch and lesson filename are omitted, a student report is generated. If the optional switch is omitted, but the lesson filename is included, a lesson report is generated.

If LPT is included in a REPORT command, a lesson filename must be specified.

If the A switch (for an answer report) is used, a lesson filename must be specified.

The date argument has no effect on answer reports.

Answer reports are the same whether they are printed on a user's terminal or on a line printer.

A maximum of 200 students may be included in a single student report.

Messages

TOO MANY CHARACTERS IN LESSON NAME
The lesson filename contains more than six characters.

ILLEGAL COMMAND. TRY AGAIN.
The REPORT command is not in proper format.

ILLEGAL DEVICE NAME
The device name is not TTY, TTL, or LPT.

ILLEGAL STUDENT NUMBER
A student programmer number is not a proper octal number or is not in proper format.

ILLEGAL DATE FORMAT
One of the specified dates is not in proper format.

ILLEGAL FRAME NUMBER FORMAT
The frame numbers are not in proper format.

ILLEGAL SWITCH
The switch is not one of the letters A, S, or L, or the switch is not in proper format.

TOO MANY STUDENTS INCLUDED IN REPORT COMMAND

More than 200 students are included in the report.

NO LESSON NAME WAS SPECIFIED

The lesson filename is missing from a form of the REPORT command that requires a lesson filename.

MASTER FILE NOT FOUND

The master record file cannot be found under the user's project-programmer numbers.

ANSWER FILE NOT FOUND

The answer file for the specified lesson cannot be found under the user's project-programmer numbers.

Examples

ENTER COMMAND
*REPORT LESN3 ↓ ...summary lesson report on user's terminal

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*REPORT TTL:6250-6270 ↓ ...summary student report on line printer

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*REPORT LPT:LESN2 [50-100] ↓ ...detailed lesson report covering frames 50-100 (line printer)

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*REPORT LPT:CHEM/A ↓ ...answer report on line printer

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ENTER COMMAND
*REPORT LPT:PHYS1/S ↓ ...student report on line printer

3-25
3.7.1 The MONITOR Command

Function

The MONITOR command is used to exit from Learn to monitor level.

Format

MONITOR

There are no arguments for this command.

Characteristics

After returning to monitor level, the CONTINUE command may not be used to continue execution of the Learn program.

Example

ENTER COMMAND
*MONITOR

EXIT
...user is at monitor level

3.7.2 The KJOB Command

Function

The KJOB command is used to exit from the Learn program and to begin the log-out procedure. This command is equivalent to the monitor KJOB command.

Format

KJOB

There are no arguments for this command.

Characteristics

The KJOB command runs the KJOB program. The KJOB program types the word CONFIRM: on the user's terminal. At this point, all of the normal options are available to the user.

Example

ENTER COMMAND
3.7.3 The BYE Command

Function

The BYE command is used to log-out directly without running the KJOB program.

Format

BYE

There are no arguments for this command.

Characteristics

The BYE command runs the LOGOUT program that logs-out the user directly without deleting any disk files.

Example

ENTER COMMAND

*BYE

JOB 29, USER 2400,2400 LOGGED OFF TTY25 1425 17-JUN-74
SAVED ALL FILES (440 BLOCKS)
RUNTIME 0 MIN, 03.06 SEC CONNECT TIME: 8 MIN. 06.00 SEC
APPROXIMATE COST OF RUN = $ 0.73
The Learn Lesson Editor is a program in the Learn system that has been specifically designed for editing Learn lesson files. Using the Lesson Editor, lesson authors can enter a new lesson into the computer system and store it on the disk, and they can make changes in lessons that were entered previously. Only unprocessed lessons can be edited, and the unprocessed lesson file must be stored on the disk under the project-programmer numbers of the user who is doing the editing. Ordinarily, other editing programs, such as TECO, should not be used because the Learn Lesson Editor automatically inserts special control characters into the lesson that are used by the Lesson Processor in processing the lesson; other editors do not insert these characters into the lesson.

4.1 Entering the Learn Lesson Editor

The three ways of entering the Lesson Editor are by means of 1) a WRITE command, 2) an EDIT command, and 3) a test mode :E command. When the WRITE command is used, the edit command A (for Append) is implied so that the lesson author can begin to type in the new lesson immediately. When the EDIT command or the test mode :E command is used, the Lesson Editor types a > sign to indicate that it is ready to accept a command. At this point the user may type in any of the edit commands that are described in section 4.3.

4.2 Line Numbers

The Lesson Editor automatically numbers each line before typing it, and it also numbers lines as they are typed in by the lesson author (see section 2.1). These line numbers are not permanent, and they do not become part of the unprocessed lesson file. Because the line numbers are not permanent, care must be exercised when several changes are made in a single frame. For example, suppose that the 10th line in some frame is to be deleted and that a line is to be inserted in front of the 5th line in the same frame. If the line is inserted first, then the numbers of all lines from line 5 on will be increased by one, and in order to delete what was formerly line 10, it would be necessary to delete line 11 instead. This difficulty can often be avoided by inserting and deleting lines in order of decreasing line number. In the preceding example, line 10 could have been deleted before inserting the line in front of line 5.
4.3 Learn Edit Commands

Learn edit commands allow lesson authors to append, delete, insert, and replace selected lines or frames in a lesson. Edit commands are terminated by a carriage return, and they may contain any number of extra spaces. The commands are described below in alphabetical order.

4.3.1 The A Command

The A command is used to append lines onto the end of a frame or frames onto the end of a lesson. The two forms of the A command are:

1) A
2) An

The first form is used to append frames onto the end of a lesson. The frame number that is assigned to the first appended frame is equal to the smallest integral multiple of the frame increment, which is larger than the frame number of the last frame in the lesson. For example, if the frame increment is 10, and the number of the last frame is 99, then the number of the first frame appended to the lesson would be 100. An append command may not be used if it results in a frame number greater than 999.

The second form of the A command is used to append lines onto the end of the frame with frame number n. The Lesson Editor types line numbers on the user's terminal before each line is typed in. After all lines have been appended the append command is terminated by typing the ESC (or altmode) key. A single frame may not contain more than 64 lines. The example below shows how lines can be appended to a frame which is assumed to contain 18 lines.

```
> A400 ↓
19 > D E ↓
20 > F↓
21 > @ ↓
22 > F↓
23 > A↓
24 > $

...user types append command
...user adds lines to frame
...user types ESC to stop appending lines

>

...Learn is ready to accept the next command
```

4-2
4.3.2 The C Command

The C command is used to copy a frame which has already been entered into a lesson. This command is useful when two or more very similar frames are required in a lesson. The form of the C command is:

\[ C_{n,m} \]

where \( n \) is the number of the frame being copied and \( m \) is the frame number assigned to the new frame. The new frame number must be greater than zero and less than 1000, and it may not be equal to the frame number of an already existing frame. In the example below, frame 30 is copied and assigned the frame number 650.

\[ C_{30,650} \]

...user types copy command

\[ \]

...Learn is ready to accept the next command

4.3.3 The D Command

The D command is used to delete selected frames or lines from a lesson. It may not be used to delete an entire lesson file (use the DELETE command for this). The four forms of the D command are:

1) \( D_{n} \)
2) \( D_{n-m} \)
3) \( D_{n,p} \)
4) \( D_{n,p-q} \)

The first form is used to delete frame \( n \) from the lesson file, and the second form is used to delete frames \( n \) through \( m \) from the lesson file. The third form is used to delete line \( p \) from frame \( n \), and the fourth form is used to delete lines \( p \) through \( q \) from frame \( n \). An example is shown below.

\[ D_{190,13-15} \]

...user types command to delete lines 13-15 of frame 190

\[ \]

...Learn is ready to accept the next command

4.3.4 The I Command

The I command is used to insert frames into a lesson or lines into
a frame. This command cannot be used to append frames onto the end of a lesson or lines onto the end of a frame. The two forms of the \_!\_ command are:

1) \_In
2) \_In,p

The first form is used to insert a frame with frame number \( n \) into a lesson. The frame number must be greater than zero and less than 1000, and it may not be equal to the frame number of an already existing frame.

The second form is used to insert one or more lines in front of line \( p \) in frame \( n \). The line number \( p \) must be greater than 1 and less than 65. The Lesson Editor types line numbers on the user's terminal before each line is typed in. After all lines have been inserted, the insert command is terminated by typing the ESC (or altmode) key. A single frame may not contain more than 64 lines. The example below shows how lines can be inserted in front of line 5 in frame 60.

\[ \_\text{I60,5} \]  
\[ 5 \text{ >B - WASHINGTON} \]  
\[ 6 \text{ >C - OREGON} \]  
\[ 7 \text{ >5} \]  
\[ \_\text{>5} \]  

\[ \text{...user types insert command} \]  
\[ \text{...user inserts lines} \]  
\[ \text{...user types ESC to terminate insertion} \]  
\[ \_\text{>5} \]  
\[ \text{...Learn is ready to accept the next command} \]

4.3.5 The \_K\_ Command

The \_K\_ command is used to add a constant to all or some of the frame numbers in a lesson. A negative constant may be used, but the resulting frame numbers may not be less than or equal to zero. The two forms of the \_K\_ command are:

1) \_K\_n
2) \_K\_n,m

The first form is used to add a constant to all of the frame numbers in the lesson. No frame number may exceed 999. The second form is similar to the first form, except that the constant is added only to frame numbers greater than or equal to \( m \). An example is shown below.

\[ \_\text{K10,100} \]  
\[ \_\text{>K10,100} \]  

\[ \text{...user types K command} \]

4-4
4.3.6 The \textbf{N} Command

The \textit{N} command is used to renumber a frame. The form of the \textit{N} command is:

\texttt{Nm,n}

where \textit{m} is the old frame number and \textit{n} is the new frame number. New frame numbers must be greater than zero and less than 1000, and they must not be equal to any other frame number in the lesson. A renumbered frame is placed in proper sequence in the lesson. In the example below, frame 30 is changed to frame 45. The renumbered frame would be placed between frame 40 and frame 50.

\begin{verbatim}
>N30,45

...user types \textit{N} command

> ...Learn is ready to accept the next command
\end{verbatim}

4.3.7 The \textbf{R} Command

The \textit{R} command is used to replace selected frames, lines, or character strings in a lesson. The four forms of the \textit{R} command are:

1) \texttt{Rn}
2) \texttt{Rn,p}
3) \texttt{Rn,p-q}
4) \texttt{Rn,p,old-string,new-string}

The first form is used to replace frame \textit{n}, the second form is used to replace line \textit{p} in frame \textit{n}, and the third form is used to replace lines \textit{p} through \textit{q} in frame \textit{n}. The Lesson Editor types line numbers before each new line that is typed in by the user. Only the letter indicating the frame type may be replaced in the first line in a frame. An example \textit{R} command is shown below.

\begin{verbatim}
>R250,12

...user types replace command

12 > ANSWER 30

...user types new line 12

> ...Learn is ready to accept the next command
\end{verbatim}

The fourth form of the \textit{R} command is used to replace the old character string in line \textit{p} of frame \textit{n} by a new character string. This form may not be used to alter the first line in a frame. The old string should not contain more than 14 characters, but the new string may contain any number of characters and need not contain the same number of characters as the old string. If the
old string occurs more than once in line \( p \), only the characters in the first occurrence of the old string are replaced. A new string can be inserted at the beginning of a line if the old string contains no characters as shown in the example below.

\[ > \text{R800,15,}, \text{AND} \downarrow \]  
...user types command to insert the word AND at the beginning of line 15

\[ > \]  
...Learn is ready to accept the next command

The example below shows how a typing error can be corrected.

\[ > \text{R750,5,THI,THE} \downarrow \]  
...user types command to replace THR with THE

\[ > \]  
...Learn is ready to accept the next command

4.3.8 The T Command

The T command is used to type out selected frames or lines in a lesson. The five forms of the T command are:

1) T
2) Tn
3) Tn-m
4) Tn,p
5) Tn,p-q

The first form causes the entire lesson to be typed, the second form causes frame \( n \) to be typed, the third form causes frames \( n \) through \( m \) to be typed, the fourth form causes line \( p \) of frame \( n \) to be typed, and the fifth form causes lines \( p \) through \( q \) of frame \( n \) to be typed. An example T command is shown below.

\[ > \text{T300,13-15} \downarrow \]  
...user types T command

13 ANSWER
14 A + NEWTON
15 B - GALILEO

\[ > \]  
...Learn is ready to accept the next command

4-6
4.3.9 The X Command

The X command is used to change the frame increment which is normally set to ten (see section 2.1). The frame increment, which must be greater than zero, is equal to the difference between the frame numbers of successive frames which are appended to the lesson. This command is useful when more than 99 frames must be included in a single lesson since the frame number of the last frame may not exceed 999. An example X command is shown below.

```
> X5 ↓

...user types X command
```

```
> 

...Learn is ready to accept the next command
```

4.4 Exiting from the Learn Lesson Editor

There are four commands that can be used to terminate lesson editing and to exit from the Lesson Editor. Exiting from the Lesson Editor by typing "control C" may result in a loss of all or part of the lesson. The four exit commands are:

1) E
2) EP
3) ET
4) M

The first form is used to terminate editing and return to Learn command level; the second form terminates editing and processes the lesson; the third form terminates editing, processes the lesson, and begins execution of the lesson in test mode; the fourth form terminates editing and returns the user to monitor level. Example exit commands are shown below.

Example 1.

```
>E ↓

ENTER COMMAND
```

...user types E command

```
...Learn is ready to accept any of the commands described in Chapter 3
```

Example 2.

```
>EP ↓

TOTAL ERRORS DETECTED: 0
```

...user types EP command

...no errors were detected during lesson processing
Example 3.

>ET

TOTAL ERRORS DETECTED: 0

START AT FRAME

Example 4.

>M

EXIT

.

4.5 Editing Efficiency

Editing very large lesson files may require a relatively large amount of CPU time, which is quite expensive. CPU time is minimized by editing frames in order of increasing frame number. For example, frame 25 should be edited before frame 50. Also, when several changes must be made, it is better to make all changes before testing the lesson rather than testing the lesson after each change is made. Of course, it is always best to avoid making errors in the first place.

4.6 An Example of Lesson Editing

ENTER COMMAND

EDIT PHYS1

> T30,7

7 ANSWER 20

>R30,7

7 > ANSWER 30

>X2

>A

...Learn is ready to accept any of the commands described in Chapter 3

...user types ET command

...user types M command

...user is at monitor level

...user types command to edit PHYS1

...user types T command

...user replaces line 7 in frame 30

...user changes frame increment to two

...user appends frames to lesson 4-8
702 TYPE-> Q
3 > ENERGY IS THE ABILITY TO DO
4 >
5 > ANSWER
6 > PHONIC ON
7 > A + WORK
8 > ACTION
9 > A
10 > F:
11 > ?
12 > F:
13 > Q:
14 > @
15 > F:
16 > A:
17 > $ 

...user types ESC to end the frame

704 TYPE-> Q
3 > WORK IS EQUAL TO FORCE TIMES
4 >
5 > ANSWER
6 > A - DISTANCE
7 > B - MASS
8 > ACTION
9 > A
10 > F:
11 > B
12 > F: NO, IT'S NOT MASS.
13 > WORK = FORCE X DISTANCE
14 > C
15 > F:
16 > A:
17 > $ 

...user types minus sign by mistake

706 TYPE-> Q

...user continues to add frames to the lesson

712 TYPE-> $ 

...user types ESC key to stop appending frames

>ET 

...user types ET command

TOTAL ERRORS DETECTED: 0

4-9
START AT FRAME*702↓

FM.702

ENERGY IS THE ABILITY TO DO*WORK↓.

VERY GOOD!

FM.704

WORK IS EQUAL TO FORCE TIMES*DISTANCE↓.

NOT SO!

ANS: WORK

...since frame 704 contains no right answer, the answer from the preceding frame is printed

FM.706

WORK IS A*:E↓ QUANTITY. ...user decides to edit the lesson

>R704,6,-,+↓ ...user replaces "-" with "+

>ET↓ ...user types ET command

TOTAL ERRORS DETECTED: 0

START AT FRAME*704↓

FM.704

WORK IS EQUAL TO FORCE TIMES*DISTANCE ↓.

RIGHT!

FM.706

WORK IS A*:→ QUANTITY. ...user terminates the lesson

ENTER COMMAND * ...Learn is ready to accept any of the commands described in Chapter 3

4-10
4.7 Summary of Learn Edit Commands

1) A - append frames
2) An - append lines

1) Cn,m - copy frame n

1) Dn - delete frame n
2) Dn-m - delete frames n thru m
3) Dn,p - delete line p
4) Dn,p-q - delete lines p thru q

1) In - insert frame n
2) In,p - insert in front of line p

1) Kn - add n to all frame nos.
2) Kn,m - add n to all frame nos. starting at frame m

1) Nm,n - renumber frame m to n

1) Rn - replace frame n
2) Rn,p - replace line p
3) Rn,p-q - replace lines p thru q
4) Rn,p,old-string, new-string - replace old string

1) T - type entire lesson
2) Tn - type frame n
3) Tn-m - type frames n thru m
4) Tn,p - type line p
5) Tn,p-q - type lines p thru q

1) Xn - change frame increment to n

1) E - exit from Lesson Editor
2) EP - exit and process lesson
3) ET - exit and test lesson
4) M - return to monitor level
CHAPTER 5

THE LEARN INSTRUCTIONAL LANGUAGE

Learn lessons are programmed in LIL, the Learn Instructional Language. LIL is both easy to learn and easy to use, and it should be possible for persons with little programming experience to master the basics of LIL programming in a matter of a few hours. However, as with any other programming language, it is necessary that the rules for writing program statements be strictly followed.

LIL is an interpretive language. After a lesson has been written in LIL, it is processed by the Lesson Processor. The processed lesson is not able to run by itself, but it is interpreted by another computer program that is called the Learn Lesson Interpreter. The Lesson Interpreter scans each statement in a lesson, and then it performs the functions necessary to execute that statement.

5.1 LIL Frames

LIL lessons are composed of basic units called frames. The first line in each frame contains the frame number and the frame type. The format of the first line in each frame is as follows:

n TYPE-t

where n is the frame number, and t is a letter that indicates the frame type. The frame number must be an integer in the range 1-999. Frames are numbered automatically by the Lesson Editor, usually beginning with ten and incrementing by ten. The frame type t may be any one of the letters Q, C, R, or B. Q-frames (Question frames) are used to present questions to students and to analyze their responses. C-frames (Calculation frames) are used to perform mathematical calculations. R-frames (Review frames) are used to set review points in a lesson, and B-frames (Break frames) are used to establish points at which students may quit working on a lesson and return to that point at a later time. A maximum of 64 lines may be contained in a single frame. The programming statements that may be used in each type of frame are described below.

5.2 Q-frames

The first line in a Q-frame has the following format

n TYPE-Q

5-1
where \( n \) is the frame number.

Q-frames are composed of three distinct sections with each section having its own set of programming statements. Generally, statements that are legal in one section of a Q-frame are not legal in another section. The first section of a Q-frame, the TEXT section, presents information or questions to students. The second section, the ANSWER section, inputs a student's answer and compares it with possible answers which have been anticipated by the lesson author. The third section, the ACTION section, performs actions that depend on the response given by the student, such as providing corrective feedback. All three sections of a Q-frame need not be present. However, any sections that are present must appear in the order in which they were discussed above.

5.2.1 The TEXT Section

The statements that may appear in the TEXT section of a Q-frame are described below.

5.2.1.1 The TEXT statement

TEXT statements are used to print questions or information on student terminals. Any number of TEXT statements may be included in a single Q-frame. The forms of the TEXT statement are

\[
\begin{align*}
(a) & \quad \text{TEXT text } \\
(b) & \quad \text{TEXT$ text } \\
\end{align*}
\]

where "text" may consist of one or more lines of information. The \( \backslash \) serves as the text delimiter. If form (a) is used, a carriage return followed by a line feed is printed before the text; form (b) does not print these characters. An example TEXT statement is shown below.

\begin{verbatim}
TEXT
WHAT IS THE CAPITAL OF MICHIGAN?
\end{verbatim}

The spacing of lines within the text can be controlled using the following characters at the end of a line. These characters do not act as text delimiters.

\[
\begin{align*}
(a) & \quad \backslash \leftarrow \\
(b) & \quad \backslash \leftarrow n
\end{align*}
\]

The letter \( n \) represents a positive integer that determines the
number of line feeds that are printed following the line. If the control characters shown in form (a) are used, the carriage return at the end of the line is not followed by a line feed. Form (a) is useful for constructing fill-in-the-blank questions. An example is shown below.

```
TEXT
THE CAPITAL OF * IS LANSING.←
```

The TEXT statement in this example would result in the following typeout on the student's terminal.

```
THE CAPITAL OF* IS LANSING.
```

The student types his answer in the blank following the asterisk.

TEXT statements may contain variables whose values are typed on the user's terminal. The variable name along with its format specification is placed between back slashes within the TEXT statement. (See section 5.3.2 for a discussion of format specifications and variable names.) An example is shown below.

```
TEXT
HOW MUCH IS \ A [1.0]+ \ B [1.0]?\n```

If the value of A is 6 and the value of B is 2, the following text would be typed on the student's terminal.

```
HOW MUCH IS 6 + 2?
```

Special control characters may also be transmitted to the student's terminal by means of TEXT statements. The list of control characters is placed within back slashes separated by commas. This list consists of the octal codes of the control characters. The effect that a control character has depends upon the type of terminal being used by the student. The example below shows how to ring the bell on a teletype.\(^1\)

```
TEXT
THIS STATEMENT \07\ RINGS THE BELL ON A TELETYPE.
```

5.2.1.2 The SLIDE Statement

SLIDE statements are used to control slide projectors that have been interfaced with student terminals (see Appendix A). Any

\(^1\)Teletype is a trademark of Teletype Corporation.
number of SLIDE statements may appear in a single Q-frame. The forms of the SLIDE statement are shown below.

(a) SLIDE ON  
(b) SLIDE OFF  
(c) SLIDE n  
(d) SLIDE n,m

Form (a) turns on the projector bulb, and form (b) turns it off. Form (c) moves the projector to slide n where n is an integer in the range 0-90. Form (d) is similar to form (c) except that the Lesson Interpreter waits for m seconds before going on to the next statement in the frame. Form (d) is useful for showing a series of slides in a single frame. The waiting time m may not exceed 90 seconds. The following example shows how SLIDE statements may be used in connection with TEXT statements.

```
TEXT
EXAMINE THE TABLE ON SLIDE 5\nSLIDE 5
SLIDE ON
TEXT
ACCORDING TO THIS TABLE, WHAT IS THE HARDNESS OF TALC?\n```

5.2.1.3 The PAGE Statement

The PAGE statement is used to erase the screen of CRT display terminals. In order for the PAGE statement to have the desired effect, the value of the special variable $D$ should be set to the device number of the type of terminal being used by the student. (See section 5.3.2.5 for a discussion of special variables). The form of the PAGE statement is

```
PAGE
```

5.2.1.4 TEXT Section Switches

There are a number of statements that can be used to set Learn switches. The switches control certain automatic features of Learn such as record keeping and access to the calculation mode. A switch statement consists of a switch name followed by one of the arguments listed below.

(a) ON  
(b) OFF  
(c) ON>  
(d) OFF>
Form (a) turns on the switch for the frame in which the statement occurs, whereas form (b) turns the switch off for the frame in which the statement occurs. Form (c) turns on the switch for the frame in which the statement occurs and for all following frames until another switch statement is encountered. Form (d) is similar to form (c) except that the switch is turned off rather than on. Example switch statements are shown below.

```
CALC OFF
STORE ON >
```

A discussion of switch statements that may appear in the TEXT section of Q-frames follows.

5.2.1.4.1 The CALC Statement

When the CALC switch is on students may enter the calculation mode by typing the ESC key, but when the CALC switch is off students cannot enter calculation mode. If a student attempts to enter calculation mode when the CALC switch is off, the following message is typed on his terminal.

```
CALCULATION MODE NOT AVAILABLE. PLEASE ANSWER THE QUESTION.
*
```

The CALC switch is automatically turned on at the beginning of a lesson.

5.2.1.4.2 The HINT Statement

When the HINT switch is on students may request a hint by typing a question mark instead of answering the question. Lesson authors should include hint actions in the ACTION section of the Q-frame. If a student requests a hint when the HINT switch is off, the following message is typed on his terminal.

```
NO HINTS AVAILABLE. PLEASE ANSWER THE QUESTION.
*
```

The HINT switch is automatically turned off at the beginning of a lesson.

5.2.1.4.3 The REVIEW Statement

When the REVIEW switch is on students may review parts of a lesson by typing a number sign (#) followed by a topic number. (See
section 5.4 for information on establishing review points in a lesson). If student requests a review when the REVIEW switch is off, the following message is typed on his terminal.

REVIEW FEATURE NOT AVAILABLE. PLEASE ANSWER THE QUESTION.
*

The REVIEW switch is automatically turned off at the beginning of a lesson.

5.2.1.4.4 The RECORD Statement

When the RECORD switch is on, two words of student performance data are stored for each frame that is executed. These data are not stored if RECORD is off. RECORD is automatically turned on at the beginning of a lesson.

5.2.1.4.5 The STORE Statement

When the STORE switch is on, student answer strings are recorded in an answer file which has been established for the lesson. Answer strings are not stored if STORE is off. STORE is automatically turned off at the beginning of a lesson.

5.2.1.4.6 The BOTH Statement

When BOTH is on, both hint labels and unanticipated answer labels are considered equivalent. This is useful if the same feedback is supplied for hints and unanticipated answers. If BOTH is on and the student types an unanticipated answer followed by a hint request, then the actions following the second hint label in the ACTION section will be executed rather than the actions following the first hint label. This prevents the student from receiving the same feedback two times in a row. BOTH is automatically turned off at the beginning of a lesson.

5.2.2 The ANSWER Section

The statements which may appear in the ANSWER section of a Q-frame are discussed below.

5.2.2.1 The ANSWER Statement

The ANSWER statement tells the Lesson Interpreter to wait for the student to type in an answer. The four forms of the ANSWER
statement are

(a) ANSWER
(b) ANSWER*
(c) ANSWER n
(d) ANSWER* n

where n is the number of seconds that the Lesson Interpreter waits for the answer. The wait time must be a positive integer less than or equal to 8100. Normally, an asterisk is printed on the student's terminal to indicate that an answer is expected; however, printing of the asterisk is suppressed if either form (b) or form (d) is used. ANSWERS may be used instead of ANSWER.

5.2.2.2 Anticipated Answers

It is often possible to anticipate several correct and incorrect answers that a student might type in to a question. These answers are listed following the ANSWER statement according to one of the formats shown below.

(a) alphabetic-label indicator answer-string
(b) numeric-label indicator expression
(c) numeric-label indicator expression, units

An alphabetic-label consists of one of the letters A-Z, and a numeric-label consists of one of the digits 0-9. The label is used to identify the answer for the ACTION section of the Q-frame. The indicator consists of a + for a correct answer, a - for an incorrect answer, or a 0 for a neutral answer. The expression is a mathematical expression which is constructed according to the rules that are discussed in section 5.3.4. Format (a) is used for answers consisting of ordinary words, while formats (b) and (c) are used for numeric answers. Form (c) allows units, such as meter or kilograms, to be included in the answer. Example anticipated answer lists follow.

ANSWER
A + WASHINGTON
B - LINCOLN

ANSWER
1 + 24, POUNDS
2 - 24

Notice that the right-wrong-neutral indicator must be set off by a space on both sides. The same label may be used for more than one answer. For example, there may be two answers labelled A.
5.2.2.3 Unanticipated Answers

Unanticipated answers are normally considered incorrect. This can be changed by placing a question mark label (?) at the end of the answer list followed by a plus sign or a zero. An example follows.

```
ANSWER
A + WASHINGTON
B - LINCOLN
? 0
```

5.2.2.4 Zero Labels

A zero label indicates that a C-frame statement follows (section 5.3). One important use of the zero label is to set limits on the acceptable error of a numeric answer. This is done by setting the value of the special variable $\%$ to the maximum allowable error as shown in the example below.

```
ANSWER
0 $\%=2.5
1 + 25, KILOGRAMS
2 0 25
```

In the above example, a student's answer would be correct if it was in the range 22.5 to 27.5.

5.2.2.5 The MATCH Statement

MATCH statements are placed in the list of anticipated answers in order to indicate that only some of the words in the following answer need to be found in the student's answer in order for a match to occur. The form of the MATCH statement is

```
MATCH n
```

where $n$ is a positive integer that indicates the number of words in the lesson answer that must be found in the student's answer. This is illustrated in the following example.

```
ANSWER
MATCH 2
A + RED WHITE BLUE
```

In this example, a student's answer would be correct if it contained only two of the three words RED, WHITE, and BLUE.
5.2.2.6 The DELIM Statement

A word in an anticipated answer is defined as any string of one or more characters located between characters in the list of word delimiters. The Lesson Interpreter sets up a list of standard word delimiters at the beginning of a lesson. These delimiters are listed below.

- space
- period
- comma
- semicolon
- colon
- exclamation point
- question mark

The DELIM statement is used to replace the list of delimiters. The two forms of the DELIM statement are

- (a) DELIM @list
- (b) DELIM

where "list" is a list of characters that replaces all the characters in the current list of delimiters. Form (b) restores the list of standard delimiters. An example is shown below.

```
DELIM @ ?!.
```

This DELIM statement places a space, question mark, exclamation point, and a period in the delimiter list.

5.2.2.7 Answer Comparison Switches

There are three switch statements that may appear in the list of anticipated answers to control the procedure used in matching lesson answers and student answers. The format of the switch statements is exactly the same as for the switches discussed in section 5.2.1.4.

5.2.2.7.1 The EXTRA Statement

When the EXTRA switch is on, extra words are allowed in a student answer that do not appear in the lesson answer. This is illustrated in the following example.

```
ANSWER
EXTRA ON
A + WASHINGTON
```

5-9
In this example, THE FIRST PRESIDENT WAS WASHINGTON would be a correct answer even though it contains words other than WASHINGTON.

The EXTRA switch is turned on automatically at the beginning of a lesson.

5.2.2.7.2 The ORDER Statement

When the ORDER switch is on, a student's answer will match a lesson answer only if the words in both answers are in the same order. When the ORDER switch is off, a match can occur even if the words in the two answers are not in the same order. The ORDER switch is turned off automatically at the beginning of a lesson.

5.2.2.7.3 The PHONIC Statement

When the PHONIC switch is on, a word in a student's answer will match a word in a lesson answer if the two words sound alike. This feature is used to accommodate minor misspellings on the part of students. This switch is turned off automatically at the beginning of a lesson. The procedure used in making the phonetic comparison is described in Appendix B. The user is warned that this procedure is very crude.

5.2.3 The ACTION Section

The statements which may appear in the ACTION section of a Q-frame are described below.

5.2.3.1 The ACTION Statement

The ACTION statement tells the Lesson Interpreter that the ACTION section of a Q-frame has begun. Its form is

\[
\text{ACTION}
\]

ACTIONS may be used in place of ACTION.

5.2.3.2 The F: Statement

The F: statement is used to provide feedback to a student by typing a message on his terminal. The forms of the F: statement are

(a) F:
(b) F:text

5-10
where "text" consists of one or more lines of information. The text in an F: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). If form (a) is used, a randomly selected comment is printed on the student's terminal which informs him whether his answer is correct or incorrect. No message is printed in the case of a neutral answer. An example is shown below.

F: YOUR ANSWER IS INCORRECT.
EINSTEIN ORIGINATED THE THEORY OF RELATIVITY.

5.2.3.3 The A: Statement

The A: statement is used to print the correct answer on a student's terminal. The two forms of the A: statement are

(a) A:
(b) A:text

where "text" consists of one or more lines of information. The text in an A: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). Form (a) prints ANS: in front of the correct answer, while form (b) prints the "text" in front of the correct answer. An example follows.

A: THE CORRECT ANSWER IS

5.2.3.4 The R: Statement

The R: statement is used to repeat a Q-frame beginning at the ANSWER statement. The two forms of the R: statement are

(a) R:
(b) R:text

where "text" consists of one or more lines of information. The text in an R: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). If the student gives an unanticipated, neutral answer, form (a) prints the message "NO SUCH CHOICE. CHOOSE ONE OF THE ABOVE LETTERS." on the student's terminal. This feature is intended to be used for multiple choice questions in which the student responds with something other than one of the letters that label the choices. If the student gives an incorrect answer, form (a) prints a randomly selected comment such as "NOT SO! TRY AGAIN." Form (b) prints the "text" on the student's terminal. After typing the message, the Lesson Interpreter returns to the ANSWER statement and waits for the student to type in another answer. An example follows.

5-11
R: USE THE FORMULA F=MA. TRY AGAIN.

5.2.3.5 The Q: Statement

The Q: statement is used to repeat an entire Q-frame from the beginning. In other respects, it is the same as an R: statement. Q: is used when it is desirable to print the question over before requiring the student to type another answer. The two forms of the Q: statement are

(a) Q:
(b) Q:text

where "text" consists of one or more lines of information. The text in a Q: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). An example is shown below.

Q: PLEASE TRY AGAIN. REMEMBER THAT THE ACCELERATION OF AN OBJECT IS DIRECTLY PROPORTIONAL TO THE FORCE ON IT.

5.2.3.6 The B: Statement

The B: statement is used to branch to a specified frame. The form of the B: statement is

B:n

where n is the frame number. The example below shows how to branch to frame 500.

B:500

5.2.3.7 The G: Statement

The G: statement is used to call subroutines and to return from them. A subroutine is a series of frames that can be called from any point in the lesson. After the series of frames has been executed, the Lesson Interpreter returns to the frame that immediately follows the frame from which the subroutine was called. The two forms of the G: statement are

(a) G:n
(b) G:

where n is the frame number of the first frame in the subroutine. Form (a) is used to call a subroutine, while form (b) must be used

5-12

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
at the end of the subroutine in order to return to the main part of the lesson.

5.2.3.8 The C: Statement

The C: statement is used to perform mathematical calculations within a Q-frame. The form of the C: statement is

\[
C:\text{statement}
\]

where "statement" is a C-frame statement (see section 5.3). An example is shown below.

\[
C:N=N+1
\]

5.2.3.9 The P: Statement

The P: statement is used to erase the screen of a CRT terminal. The form of the P: statement is

\[
P:
\]

In order for P: to have the desired effect, the value of the special variable $D$ should be set to the device number for the type of terminal being used by the student. (See section 5.3.2.5 for a discussion of special variables).

5.2.3.10 The S: Statement

The S: statement is used to control slide projectors that have been interfaced with student terminals (see Appendix A). The four forms of the S: statement are given below.

(a) \(S:\text{ON}\)
(b) \(S:\text{OFF}\)
(c) \(S:n\)
(d) \(S:n,m\)

The four forms are equivalent to the four forms of the SLIDE statement (see section 5.2.1.2). An example of how to move the projector to slide 50 is shown below.

\[
S:50
\]
5.2.3.11 The E: Statement

The E: statement is used to end lesson execution. The two forms of the E: statement are

(a) E:
(b) E: text

where "text" consists of one or more lines of information. The text of an E: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). If form (a) is used, the lesson is simply terminated, but if form (b) is used, the "text" is typed on the student's terminal before the lesson is terminated. An E: is not required at the end of a lesson since the lesson is terminated automatically at this point. An example follows.

E: CONGRATULATIONS! YOU SEEM TO HAVE MASTERCED THE MATERIAL IN THIS LESSON. THERE IS NO NEED TO GO ANY FURTHER. SEE YOU LATER.

5.2.3.12 The L: Statement

The L: statement is used to end a lesson and log the student off the computer system. The two forms of the L: statement are

(a) L:
(b) L: text

where "text" consists of one or more lines of information. The text in an L: statement has the same features as the text in a TEXT statement (see section 5.2.1.1). If form (a) is used, the student is logged off immediately, but if form (b) is used, the "text" is printed on the student's terminal before log-out begins. An example is shown below.

L: THIS IS THE END OF THE LESSON. SEE YOU LATER.

5.2.3.13 Action Labels

Action labels are placed in the ACTION section of a Q-frame in order to make the execution of the action statements dependent upon a student's answer. Action labels may consist of single letters or digits, or a label may consist of one of the special characters that will be discussed later. Several labels may be placed on the same line separated by spaces or commas, and the same label may appear more than once in the same frame.
After a student has typed in his answer, the Lesson Interpreter scans the list of anticipated answers until a matching lesson answer is found or until the ACTION statement is encountered. If a matching lesson answer is found, the Interpreter stores the answer label and goes directly to the ACTION section. The Interpreter then scans the ACTION section until it encounters an action label that is the same as the answer label. The action statements following this label are then executed. If the student's answer does not match any of the lesson answers, the Interpreter searches for a question mark (?) label, and executes the statements that follow it. An example is shown below.

```
ACTION
A
F:
B
R: YOU'RE CLOSE, BUT TRY AGAIN.\n?
F:
A:
```

In this example, the first F: statement would be executed if the student's answer matched the lesson answer that was labelled A. The Lesson Interpreter would then skip the rest of the ACTION section and go to the next frame. Similarly, the R: statement would be executed only if the student's answer matched the answer labelled B. If the student's answer did not match any of the lesson answers, the F: and the A: would be executed. Notice that action statements and action labels may not appear on the same line.

In cases where a student has repeated a frame and has given the same answer more than once, the Lesson Interpreter does not execute the action statements following the first action label that is the same as the answer label. Instead it searches for the action label that corresponds to the number of times that particular answer has been repeated by the student. For example, if the answer labelled B has been repeated three times, the Interpreter searches for the third occurrence of the B label in the ACTION section and executes the action statements following it.

If a student has given the same answer three times, and there are only two action labels for that answer, then the Interpreter will go on to the next frame without performing any actions. This can be prevented by using the @ label. Actions following this label are always executed. For this reason, the @ label is normally the last action label in a frame, and the action statements following it normally give the student the correct answer. This is shown in the following example.

```
ACTION
A
```

5-15
In this example, if the student gave an unanticipated answer, he would be given a hint (USE F=MA) and told to try again. If he then gave another unanticipated answer, the action statements following the @ label would be executed so that the student would be given the correct answer. The same thing would happen if the student gave answer B two times in a row. However, if the student gave answer B followed by an unanticipated answer, he would receive the same hint twice!

5.2.3.13.1 The Time Up Label

If a student does not respond to a question in the time specified in the ANSWER statement, the Lesson Interpreter goes on to the ACTION section and searches for the ^ label. If one is found, the action statements following it are executed. An example is shown below.

ACTION
A
^ 
F:YOUR TIME IS UP. PLEASE TRY TO RESPOND MORE QUICKLY.
A:THE ANSWER WAS
@ 
F: 
A:

5.2.3.13.2 Hint Labels

If a student requests a hint by typing a question mark, and if the HINT switch is on (see section 5.2.1.4.2), the Lesson Interpreter goes directly to the ACTION section and searches for a hint label. A hint label consists of the number sign (#) character unless it is the last hint label in a frame. The last hint label should be an exclamation point (!). The first time a hint is requested, the action statements following the first hint label are executed. If a second hint is requested, the action statements following the second hint label are executed, and so on. If the student requests more hints than are available, the action statements following the last hint label (!) are executed.
The procedure described above is modified slightly when the BOTH switch is on (see section 5.2.1.4.6). In this case, an unanticipated answer is counted as a hint request, and vice versa. When the BOTH switch is on, and a student gives an unanticipated answer followed by a hint request, the action statements following the second hint label are executed instead of the action statements following the first hint label. An example follows.

ACTION
1
F:
# ?
F:
R:TRY USING THE FORMULA $F=MA$.
/ @
F:
A:

In this example, if the BOTH switch was on and if a student requested a hint and then gave an unanticipated answer, he would not receive the message \textit{TRY USING THE FORMULA $F=MA$} twice. This is because the hint would be counted as an unanticipated answer.

5.2.3.14 Unconditional Actions

If action statements are placed directly under the ACTION statement with no action labels preceding them, they will be executed regardless of what answer was given by the student. After executing such statements, the Lesson Interpreter searches for the action label corresponding to the student's answer and executes the statements following it. An example is shown below.

ACTION
C:N=N+1
A
F:
@
F:
A:

In this example, the C: statement is an unconditional action.

5.2.4 The REM Statement

The REM statement allows lesson authors to insert remarks into a lesson for documentation purposes. A REM statement may be used in all three sections of a Q-frame. An example is shown below.
REM SLIDE 7 SHOWS AN IGNEOUS ROCK.

5.2.5 Example Q-frames

Example #1

100 TYPE-Q
TEXT
IN ANTHROPOLOGICAL TERMS, A GROUP OF PEOPLE WHO OCCUPY A
PARTICULAR TERRITORY AND SPEAK A COMMON LANGUAGE WHICH
IS NOT GENERALLY INTELLIGIBLE TO NEIGHBORING PEOPLES IS
A(AN): 2
A. ETHNIC GROUP
B. CULTURE
C. SOCIETY
D. SUBCULTURE
\ ANSWER
EXTRA OFF
A + C
MATCH 1
B - A,B,D
? 0
ACTION
A
F:
B     B
R:
B
F:
A:
@
R:

A sample execution of this frame is shown below.

IN ANTHROPOLOGICAL TERMS, A GROUP OF PEOPLE WHO OCCUPY A
PARTICULAR TERRITORY AND SPEAK A COMMON LANGUAGE WHICH
IS NOT GENERALLY INTELLIGIBLE TO NEIGHBORING PEOPLES IS
A(AN):

A. ETHNIC GROUP
B. CULTURE
C. SOCIETY
D. SUBCULTURE
*F

NO SUCH CHOICE.   CHOOSE ONE OF THE ABOVE LETTERS.

5-18
Example #2

150 TYPE-Q
SLIDE 5
SLIDE ON
TEXT
ACCORDING TO THE GRAPH ON SLIDE 5, AS THE FREQUENCY OF
THE INCIDENT RADIATION INCREASES, THE ENERGY OF THE
PHOTOELECTRON

ANSWER
EXTRA OFF
PHONIC ON
A + INCREASES
B - DECREASES
? 0
ACTION
A
F:
B: 300
B
F: NO! JUST THE OPPOSITE.
Q: PLEASE ANSWER INCREASES OR DECREASES.

A sample execution of this frame is shown below.

ACCORDING TO THE GRAPH ON SLIDE 5, AS THE FREQUENCY OF
THE INCIDENT RADIATION INCREASES, THE ENERGY OF THE
PHOTOELECTRON CHANGES.

PLEASE ANSWER INCREASES OR DECREASES.

ACCORDING TO THE GRAPH ON SLIDE 5, AS THE FREQUENCY OF
THE INCIDENT RADIATION INCREASES, THE ENERGY OF THE
PHOTOELECTRON DECREASES.

NO! JUST THE OPPOSITE.
Example #3

AN OBJECT WITH A MASS OF 100 KILOGRAMS IS ACCELERATING AT THE RATE OF 3 M/SEC/SEC. WHAT IS THE FORCE ON THE OBJECT?

HINT ON ANSWER
1 + 300, NEWTONS
1 + 300, N
2 + 300
ACTION
1 F:
2 F:
F:
R: USE F = MA

R: YOU MUST MULTIPLY 30 TIMES 10.

A sample execution is shown below.

AN OBJECT WITH A MASS OF 100 KILOGRAMS IS ACCELERATING AT THE RATE OF 3 M/SEC/SEC. WHAT IS THE FORCE ON THE OBJECT?

USE F = MA.
*300

YOUR ANSWER IS CORRECT, BUT REMEMBER THAT THE UNITS ARE NEWTONS.
5.3 C-frames

C-frames are used to perform mathematical calculations. C-frame statements may be used not only in C-frames, but also in Q-frames following a zero answer label (see section 5.2.2.4) or in a C: statement (see section 5.2.3.8). In addition, students use C-frame statements to perform mathematical calculations while they are in the calculation mode.

The first line in a C-frame has the form

```
 n TYPE-C
```

where n is the frame number. Unlike Q-frames, C-frames are not divided into distinct sections.

5.3.1 The Format of C-frame Statements

With the exception of the FILL statement (see section 5.3.5.2), C-frame statements may not occupy more than one line. Each statement consists of an optional line number, an optional iteration field, and a required statement field.

5.3.1.1 Line Numbers

The line number, if it is present, consists of an integer in the range 1-24. This line number is not the same as the line number that is supplied by the Lesson Editor (see section 2.1). Normally, only C-frame statements that are referred to in GOTO statements (see section 5.3.7.1) are numbered. Two or more lines in the same frame may not have the same line number. The example below illustrates the use of a line number in a C-frame statement.

```
 20 INPUT A,B
```

5.3.1.2 Iteration Fields

The iteration field, if it is present, causes the execution of a C-frame statement to be repeated a specified number of times. The two forms of the iteration field are

- (a) [index=limit1,limit2]
- (b) [index1=limit1,limit2 index2=limit3,limit4]

where index, index1, and index2 are regular variable names (see 5-21
section 5.3.2.3), and where limit1, limit2, limit3, and limit4 are mathematical expressions (see section 5.3.2) that may be quite complex or that may consist of only a single number of variable name.

If form (a) is used, the C-frame statement is executed a sufficient number of times so that the index variable can take on all of the integer values between and including the specified limits. When the statement is first encountered by the Lesson Interpreter, the values of the mathematical expressions, limit1 and limit2, are calculated, and the results are truncated to integers. The first time the statement is executed, the index variable takes on the value of limit1. If the value of limit2 is greater than the value of limit1, then the value of the index variable is increased by one each time the statement is executed; if the value of limit2 is less than the value of limit1, then the value of the index variable is decreased by one each time the statement is executed. At the completion of all iterations, the index variable has the value of limit2. Form (b) iteration fields are similar to form (a) iteration fields, except that for each value of the first index, the statement is repeated a sufficient number of times for the second index variable to take on all of the values specified by limit3 and limit4. Example C-frame statements that contain iteration fields are shown below.

\[
[I=1,10] \text{INPUT } AX(I)
[I=N1,N2 \ J=N3,N4] \text{TYPE } BK(I,J)
[K=AX(I),2*AX(10)] T=T+AA(K)
\]

The user may find the above examples more instructive after he has read further in this chapter.

5.3.1.3 Statement Fields

The statement field may contain any of the C-frame statements that are described later in this chapter.

5.3.2 Mathematical Expressions

Many C-frame statements may contain mathematical expressions. These expressions consist of constants, variables, operators, and functions, and they are constructed according to the rules of ordinary algebra.
5.3.2.1 Constants

Ordinary constants are stored internally by Learn as single precision, floating point numbers. Constants that appear in mathematical expressions should normally contain a maximum of seven digits. If more digits are included, there will probably be no increase in accuracy because of the manner in which constants are handled by Learn's input-output routine. Examples of ordinary constants are shown below.

2
2.678
-25
.015
0.23
15000

Constants may also be written in exponential notation. Two examples of exponential notation are shown below.

(a) 2.53E25
(b) 9.5E-12

Example (a) is a representation of the number 2.53 times ten raised to the 25th power, while example (b) represents 9.5 times ten to the minus 12th power. The absolute value of a constant must be in the approximate range 1.5E-39 to 1.7E38.

5.3.2.2 Character Constants

The characters which can be printed on a user's terminal are represented internally in the computer as seven digit, binary numbers. A character constant is equal to the value of the seven bit code of a specific character. The form of a character constant is

@character

where "character" is any character which can be printed on a terminal. Examples follow.

@A
@Z
@a
@?
5.3.2.3 Regular Variables

Mathematical expressions may contain regular variables. The name of a regular variable must consist of a single letter or of a single letter followed by a single digit. Examples of legal variable names are shown below.

A
B2
X
Y1

The value of a regular variable is stored as a single precision, floating point number, which occupies a single word in the computer's memory.

5.3.2.4 Array Variables

Mathematical expressions may also contain array variables that are either singly or doubly subscripted. Array variable names must consist of two alphabetic characters. Example array variable names are shown below.

AA
AX
BY

Before an array variable can be used in a C-frame statement, its dimensions must be declared in an ARRAY statement (see section 5.3.3.2).

When an array variable is used in a C-frame statement, the array subscripts are placed in parentheses following the array name. A subscript may consist of a number, a variable, or a mathematical expression. If an expression is used, it may not contain parentheses. In the case of a doubly subscripted variable, the two subscripts are separated by a comma. Examples are shown below.

AX(5)
BY(3,12)
AN(I,J+1)

5.3.2.5 Special Variables

There are a number of special variables that may appear in C-frame statements. In most cases, the values of the special variables are set by the Lesson Interpreter. The names of the special

5-24
variables and what they represent are listed below.

$A$ - the last numeric answer given by the student

$C$ - the total number of correct answers given by the student

$D$ - the device number of the user's terminal (automatically set to zero unless changed with an assignment statement - see Appendix F)

$E$ - the total elapsed time for the lesson

$F$ - the total number of frames executed by the student

$N$ - the total number of neutral answers given by the student

$R$ - a random number between 0 and 1 (the number is different each time $R$ is referenced)

$S$ - the last answer string typed by the student (this is a sixteen word array)

$T$ - the answering time for the last question

$U$ - the number of times that the student failed to answer within the time limit specified in an ANSWER statement

$W$ - the total number of wrong answers given by the student

$\%$ - the maximum allowable error for a numeric answer (this is set to zero at the beginning of each Q-frame)

5.3.2.6 Operators

The following operator symbols may appear in mathematical expressions.

+ for addition

- for subtraction

* for multiplication

/ for division

↑ for exponentiation

The order in which the operations are performed is as follows:

1) Any exponentiations which are present in an expression are performed first.

2) Next multiplications and divisions are performed from left to right before any additions or subtractions.
3) Finally, additions and subtractions are performed from left to right.

Parts of expressions within parentheses are evaluated before parts not inside of parentheses, and if parentheses are embedded within other parentheses, the expression within the innermost parentheses is evaluated first. Illustrative examples follow.

\[
\begin{align*}
2\times3+4 & \quad \text{(equals 10)} \\
2\times(3+4) & \quad \text{(equals 14)} \\
6/3+3 & \quad \text{(equals 5)} \\
6/(3+3) & \quad \text{(equals 1)} \\
2/(2\times2) & \quad \text{(equals 1)} \\
2/(2\times2) & \quad \text{(equals 1)} \\
(3\times(1+2))/(2+4) & \quad \text{(equals 1.5)}
\end{align*}
\]

5.3.2.7 Mathematical Functions

There are a number of mathematical functions which may be used in expressions. These functions have a single argument which is enclosed in parentheses, following the function name. The argument may be a number, a variable, or an expression. A list of the available functions is given below.

- **ABS**: absolute value
- **INT**: integer
- **SQRT**: square root
- **CBRT**: cube root
- **EXP**: exponential \(e^x\)
- **NLOG**: natural logarithm
- **LOG**: common logarithm
- **COS**: cosine (argument in radians)
- **COSD**: cosine (argument in degrees)
- **SIN**: sine (argument in radians)
- **SIND**: sine (argument in degrees)
- **TAN**: tangent (argument in radians)
- **TAND**: tangent (argument in degrees)
- **ACOS**: inverse cosine (radians)
- **ACOSD**: inverse cosine (degrees)
- **ASIN**: inverse sine (radians)
- **ASIND**: inverse sine (degrees)
- **ATAN**: inverse tangent (radians)
- **ATAND**: inverse tangent (degrees)

Example expressions that contain functions are shown below.

\[
\begin{align*}
\text{SQRT(900)+X} \\
\text{EXP(0.67*NLOG(X+Y))}
\end{align*}
\]
5.3.2.8 Special Functions

There are a number of special functions that access student performance data. These functions are described below.

5.3.2.8.1 The SEEN Function

The SEEN function returns the total number of frames that have been executed, including the frame in which the function appears. It has the following forms

(a) \text{SEEN}(f)
(b) \text{SEEN}(f&g)
(c) \text{SEEN}(f&g&n)

where \(f\) and \(g\) are frame numbers, and where \(n\) is a positive integer. Form (a) returns the number of times frame \(f\) was executed, while form (b) returns the number of times frames, with frame numbers in the range \(f\) through \(g\), have been executed. Form (c) is similar to form (b) except that the count begins with the \(n\)th execution of frame \(f\). For example, if frame 100 was executed twice, then

\text{SEEN}(100&200&2)

would return the number of frames executed by the student beginning with the second execution of frame 100.

5.3.2.8.2 The RIGHT, WRONG, and NEUT Functions

The RIGHT, WRONG, and NEUT functions return the number of right, wrong, and neutral answers given by the student. The forms of these functions are as follows.

\begin{align*}
\text{RIGHT}(f) & \quad \text{WRONG}(f) & \quad \text{NEUT}(f) \\
\text{RIGHT}(f&g) & \quad \text{WRONG}(f&g) & \quad \text{NEUT}(f&g) \\
\text{RIGHT}(f&g&n) & \quad \text{WRONG}(f&g&n) & \quad \text{NEUT}(f&g&n)
\end{align*}

The arguments of these functions are the same as those of the SEEN function (see section 5.3.2.8.1).

5.3.2.8.3 The ATIME Function

The ATIME function returns the student's answering time in seconds. The forms of the ATIME function are as follows.

\text{ATIME}(f)
The arguments are the same as for the SEEN function (see section 5.3.2.8.1).

5.3.2.8.4 The PATH Function

The PATH function returns a 1 if the sequence of frames specified in the arguments of the function were followed by the student; otherwise, it returns a zero. The form of the PATH function is

\[
\text{PATH}(f_1&f_2&f_3&...&f_n&n)
\]

where \( f_1 \) through \( f_n \) are frame numbers, and \( n \) is a positive integer which indicates that the path determination should begin with the \( n \)th execution of frame \( f_1 \). An example is shown below.

\[
\text{PATH}(50&60&100&3)
\]

This function would return the value 1 if the student had passed through frame 60 followed by frame 100 after having passed through frame 50 for the third time.

5.3.2.8.5 The TAG Function

The TAG function returns the value of the seven bit, binary code for the answer label of the answer given by the student in a specified frame. The form of the TAG function is

\[
\text{TAG}(f)
\]

where \( f \) is a frame number. TAG functions are usually used in connection with character constants (see section 5.3.2.2) as shown in the example C-frame statement that follows.

\[
\text{IF } \text{TAG}(250)=\text{B, } B:30
\]

This statement would result in a branch to frame 30 if the student had given the answer that was labelled B in frame 250.

5.3.3 Variable Definition Statements

There are two statements that cause variable names to be assigned to locations in the computer's memory. The SCALAR statement is used for regular variables, and the ARRAY statement is used for subscripted variables.
5.3.3.1 The SCALAR Statement

The form of the SCALAR statement is

```
SCALAR list
```

where "list" consists of one or more regular variable names separated by commas. An example is shown below.

```
SCALAR A,X,Y1,Y2
```

It is not necessary for regular variables to be defined in a SCALAR statement since the Lesson Processor automatically defines them as they are encountered in a lesson. However, if one lesson calls another lesson (see section 5.3.9.1) and if both of the lessons must reference the same variables, then these variables should be defined in a SCALAR statement in both lessons. The variable names should be listed in the same order in both lessons.

5.3.3.2 The ARRAY Statement

The dimensions of array variables must be defined in an ARRAY statement before the variable appears in another C-frame statement. The form of the ARRAY statement is

```
ARRAY list
```

where "list" consists of a list of variable names, along with their dimensions, separated by commas. An example is shown below.

```
ARRAY AX(10),BX(100,5)
```

A maximum of 8100 words of memory are available for variable storage.

5.3.4 Assignment Statements

An assignment statement causes the value of a mathematical expression to be assigned to a variable. The form of an assignment statement is

```
variable=expression
```

where "variable" may be a regular variable, an array variable, or one of the special variables $D$ or $. Example assignment statements are shown below.
\[ A = X + 2^*Y \]
\[ X = \text{SQRT}(A)/N \]
\[ AX(I) = BX(I) * CX(I) \]

5.3.5 Input Statements

There are two statements that may be used to input data to the computer. The INPUT statement inputs data from a user's terminal, and the FILL statement inputs data into an array from a lesson file.

5.3.5.1 The INPUT Statement

The form of the INPUT statement is

\[ \text{INPUT list} \]

where "list" consists of a list of regular and/or array variables separated by commas. When the Lesson Interpreter encounters an INPUT statement, it waits for the user to type in his data. An example follows

\[ \text{INPUT } A, X, AA(I) \]

For this statement, the user would type in three numbers separated by commas.

The following statement could be used to input a one-dimensional array of numbers.

\[ [I=1,10] \text{ INPUT } AX(I) \]

The ten numbers could all be typed on the same line separated by commas, or each number could be typed on a separate line.

A two-dimensional array of numbers could be input using this statement.

\[ [I=1,3 \ J=1,5] \text{ BX(I,J)} \]

In this case, three lines with five numbers on each line should be typed.
5.3.5.2 The FILL Statement

The two forms of the FILL statement are

(a) FILL array
data

(b) FILL name("string")

where "array" is an array variable name followed by the array dimensions in parentheses, "data" is the array of numbers which is read in, "name" is an array variable name without dimensions, and "string" is a string of alphanumeric characters.

An example of form (a) is shown below.

```
FILL AX(3,4)
1,2,25,3.5
9,7,2.6,0
2.3,8.1,0,5
```

Notice that the numbers are separated by commas.

Form (b) fills the array with a string of characters as shown in the following example.

```
FILL BB("THIS IS THE STRING.")
```

Each word in an array can hold up to five characters. The user should make sure that the string is not too long to be stored in the array variable. For example, a maximum of 25 characters can be stored in a five word array.

5.3.6 Output Statements

There are two statements that can be used to output data to a user's terminal - the TYPE statement and the ATYPE statement.

5.3.6.1 The TYPE Statement

The two simplest forms of the TYPE statement are

(a) TYPE "string"
(b) TYPE list

where "string" is a string of alphanumeric characters which are enclosed in quotation marks, and "list" is a list of mathematical
expressions which are separated by commas.

Form (a) prints the characters within the quotation marks on the user's terminal. An example is shown below.

\[ \text{TYPE "THIS IS A STRING."} \]

Form (b) prints the values of the mathematical expressions on the user's terminal. Each number is printed left-justified in a field that is fifteen characters wide. This is illustrated in the following example.

\[ \text{TYPE X+Y, SQRT(16), X*Y} \]

If \( X \) was equal to five and \( Y \) was equal to ten, the following output would appear on the user's terminal.

\[
\begin{align*}
15 & \\
4 & \\
50 & 
\end{align*}
\]

(The number 15 is at the left margin).

Forms (a) and (b) of the TYPE statement can be combined to produce a TYPE statement that contains both mathematical expressions and strings. This is illustrated in the following example.

\[ \text{TYPE "THE VALUE OF THE PH IS" LOG(H)} \]

Notice that no comma is needed between the second quotation mark and the mathematical expression \( \text{LOG}(H) \).

Format specifications may be included in TYPE statements in order to control the format in which the value of a variable or expression is printed. The forms of format specifications are

\[
\begin{align*}
(a) & \quad [f.d] \\
(b) & \quad [An]
\end{align*}
\]

where \( f \) is the total width of field in which the number is to be printed, and \( d \) is the number of digits to be printed after the decimal point. The letter "A" in form (b) indicates alphanumeric format, and \( n \) is equal to the number of characters that will be printed.

If form (a) is used, the value of the expression preceding the format specification is printed right-justified in a field \( f \) characters wide with \( d \) digits after the decimal point. If \( d \) is zero, no decimal point is printed. The field width \( f \) must be large enough to accommodate the sign of the number and a decimal point in addition to the digits. If a number is too large for the specified field width, the entire number will be printed.
beginning at the left margin of the field. In the following example, assume that \( X \) equals 2.6351 and \( Y \) equals 25.7.

\[
\text{TYPE } X[5.2],Y[7.0],\sqrt{9}[5.1]
\]

This example would produce the following output.

\[
2.64 \quad 26 \quad 3.0
\]

Notice that the numbers have been rounded off before printing.

Alphanumeric format specifications are used to print out a string of characters that has been stored under some variable name. An example is shown below.

\[
\text{TYPE } AA(1)[A15]
\]

Since only 5 characters can be stored in a single word of memory, the statement shown above would type out the contents of \( AA(1) \), \( AA(2) \), and \( AA(3) \).

Normally a carriage return and line feed follows the line of print produced by a \text{TYPE} statement. These characters can be suppressed by terminating the \text{TYPE} statement with a comma as shown in the following example.

\[
\text{TYPE } "\text{WHAT IS THE MASS OF THE OBJECT?}";
\]

This statement would allow the student to type his response on the same line as the question.

5.3.6.2 The \texttt{ATYPE} Statement

The \texttt{ATYPE} statement is used to type out arrays of numbers. The form of the \texttt{ATYPE} statement is

\[
\text{ATYPE } \text{array}
\]

where "array" consists of an array variable name followed by the array dimensions in parentheses. An example follows.

\[
\text{ATYPE } AX(3,2)
\]

This statement might result in the following output.

\[
\begin{array}{cc}
2.5 & 6.56 \\
2 & 7.8 \\
9.3 & 500
\end{array}
\]
Each number was printed left-justified in a field that is 15 characters wide.

A format specification (see section 5.3.6.1) may be used if right-justified output is desired. An example is shown below.

\[
A\text{TYPE }AX(3,2)[8.2]
\]

The following output might result from this statement.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.50</td>
<td>6.56</td>
</tr>
<tr>
<td>2.00</td>
<td>7.80</td>
</tr>
<tr>
<td>9.30</td>
<td>500.00</td>
</tr>
</tbody>
</table>

5.3.7 Branching Statements

There are three statements that may be used in C-frames to branch from one point in a lesson to another. Two of the statements, B: and G:, may also appear in Q-frames (see sections 5.2.3.6 and 5.2.3.7). The other statement is described below.

5.3.7.1 The GOTO Statement

The form of the GOTO statement is

\[
\text{GOTO line-number}
\]

where "line-number" is a line number (see section 5.3.1.1) in the same frame as the GOTO statement. An example is shown below.

\[
\text{GOTO 23}
\]

This statement would cause the Lesson Interpreter to jump to the statement numbered 23 and to continue sequential execution from that point.

5.3.8 The IF Statement

The IF statement is used to make the execution of a C-frame statement dependent upon the value of some mathematical expression. The form of the IF statement is

\[
\text{IF relation, statement}
\]

where "relation" consists of two mathematical expressions separated by a relational symbol, and "statement" is a C-frame statement.

5-34
The relational symbols that may be used are listed below.

- equals
- greater than
- greater than or equal
- less than
- less than or equal
- not equal

Some example IF statements are shown below.

IF A=B, N=N+1
IF SQRT(B) >= X+Y, GOTO 5
IF RIGHT(100&200)/SEEN(100&200) < 0.80, B:10

5.3.9 Calling External Lessons

There are two statements that allow one lesson to call another. The CALL statement is used to transfer to another lesson, and the RETURN statement is used to return to the original lesson.

5.3.9.1 The CALL Statement

The form of the CALL statement is

```
CALL lesson [p1, p2]
```

where "lesson" is the filename of the processed lesson file, p1 is the project number under which the file is stored, and p2 is the programmer number under which the file is stored. If p1 and p2 are omitted, the file is assumed to be stored under the same project-programmer numbers as the lesson that contains the CALL statement. Examples are shown below.

```
CALL MATH
CALL LESN2 [6400, 6401]
```

5.3.9.2 The RETURN Statement

The form of the RETURN statement is

```
RETURN
```

Execution of the original lesson resumes in the frame following 5-35.
the frame that contained the CALL statement. If several lessons have called each other in sequence, the RETURN is to the first lesson in the chain.

5.3.10 The RESERVE Statement

The RESERVE statement is used to reserve storage in the computer's memory for student calculation mode variables. The form of the RESERVE statement is

RESERVE n

where n is the number of words being reserved. RESER may be used in place of RESERVE.

If a RESERVE statement is not included in a lesson, ten words of storage are reserved automatically. If more than one RESERVE statement appears in a lesson, the last statement determines the number of words that will be reserved. The RESERVE statement cannot be used to change the core allocation during lesson execution.

5.3.11 The REM Statement

The REM statement may be used to insert remarks into a C-frame for documentation purposes. An example REM statement is shown below.

REM THIS FRAME GENERATES RANDOM DATA.

5.3.12 Example C-frames

Example #1

The following C-frame calculates the mean and standard deviation of a set of five numbers.

100 TYPE-C
   REM THIS FRAME COMPUTES THE MEAN AND STANDARD DEVIATION
   REM OF A SET OF FIVE NUMBERS.
   ARRAY NN(5)
   TYPE "TYPE FIVE NUMBERS SEPARATED BY COMMAS"
   [I=1,5] INPUT NN(I)
   [I=1,5] M=M+NN(I)/5
   [I=1,5] V=V+(M-NN(I))^2/4
   TYPE ""
   TYPE "MEAN ="M,"STD ="SQRT(V)
A sample execution of this frame follows.

```plaintext
TYPE FIVE NUMBERS SEPARATED BY COMMAS
1,2,3,4,5

MEAN = 3    STD = 1.581139
```

Example #2

This example shows a complete lesson that generates random data for drill problems on Newton's Second Law.

```
10 TYPE-C
IF N=2,B:30
N=N+1
A=INT(100*$R)
M=INT(100*$R+1)
F=M*A

20 TYPE-Q
TEXT
AN OBJECT WITH A MASS OF $M[1.0]$ KG ACCELERATES AT THE RATE OF $A [1.0]$ M/SEC/SEC. WHAT IS THE FORCE ON THE OBJECT?

ANSWER
1 + F, NEWTONS
1 + F, N
2 + F
ACTION
1
F:
B:10
2
F: YOUR ANSWER IS CORRECT, BUT THE UNITS ARE NEWTONS.
B:10
?
R:
@
F:
A:
B:10
```

A sample execution of this lesson is shown below.

5-37
AN OBJECT WITH A MASS OF 96 KG ACCELERATES AT THE RATE OF 42 M/SEC/SEC. WHAT IS THE FORCE ON THE OBJECT?

CALCULATION MODE

TYPE 96*42

4032

ANSWER

4032

YOUR ANSWER IS CORRECT, BUT THE UNITS ARE NEWTONS.

AN OBJECT WITH A MASS OF 4 KG ACCELERATES AT THE RATE OF 35 M/SEC/SEC. WHAT IS THE FORCE ON THE OBJECT?

140 NEWTONS

FINE!

THIS IS THE END OF THE LESSON.
SEE YOU LATER!

5.4 R-frames

The first line in an R-frame has the following format.

n TYPE-R

where n is the frame number. Any additional lines that are placed in an R-frame are typed on the student's terminal the first time that the frame is encountered.

When an R-frame is encountered for the first time, the Lesson Interpreter prints the message

TOPIC #n

where n is an integer which indicates that this is the nth review frame that has been encountered. Any text that has been placed in the R-frame is printed following this line.
After passing through an R-frame, the student can return to that point in the lesson by typing \#n where \( n \) is the number of the review frame. If he wishes to terminate the review and continue the lesson, the student may type \#C. The Lesson Interpreter automatically returns the student to the main sequence of a lesson if a review frame is encountered during the course of a review. An example R-frame is shown below.

```
100 TYPE-R
SEDMENTARY ROCKS
```

If this was the third R-frame in a lesson, the following message would be printed on the student's terminal.

```
TOPIC #3
SEDMENTARY ROCKS
```

Later on in the lesson, a student could review the discussion on sedimentary rocks by typing \#3 instead of answering a question.

5.5 B-frames

B-frames are used to establish points in a lesson at which students may quit and then return at a later time to finish the lesson. The first line in a B-frame has the form

```
n TYPE-B
```

where \( n \) is the frame number. Lines of text following the first line in a B-frame are printed on the student's terminal.

The student indicates that he wants to break away from a lesson by typing \#B. When the student wants to return to the lesson, he uses the normal GET command that he used to begin executing the lesson in the first place. Execution will resume in the frame following the last B-frame that the student passed through before typing \#B. An example B-frame is shown below.

```
250 TYPE-B
IF YOU WOULD LIKE TO TAKE A BREAK AT THIS TIME,
TYPE \#B.
```
6.1 Internal Record Keeping

The Learn Lesson Interpreter automatically stores two words of student performance data on each frame in a lesson that is executed. The information that is stored includes frame numbers, answer labels (see section 5.2.2.2), right-wrong-neutral indicators (see section 5.2.2.2), the time required to answer each question, and a record of whether or not the calculation mode was used. The storage of this information can be suppressed by means of the RECORD statement (see section 5.2.1.4.4).

NOTE: For record keeping purposes, 1000 is added to the frame numbers of all frames executed during a review (see section 5.4).

Twelve words of summary information are also stored by the Lesson Interpreter. The contents of these twelve words are listed below.

1. The lesson name
2. The student's project-programmer numbers
3. The date lesson execution began
4. The time of day lesson execution began
5. The total elapsed time for the lesson
6. The total number of correct answers
7. The total number of incorrect answers
8. The total number of neutral answers
9. The total number of time ups
10. The total answering time for the lesson
11. The total number of frames executed
12. The total CPU time for the lesson

All the information that was described above is retained in core until the completion of the lesson. At that time, the information is transferred to the transaction file TRANS.REC. The Lesson Interpreter searches for the transaction file under identical project-programmer numbers which are equal to the student's project number. If the transaction file cannot be found under these numbers, the Lesson Interpreter searches for it under the student's own project-programmer numbers. If none is found there, lesson records are not saved, and the message

LESSON RECORDS NOT SAVED

is printed on the student's terminal.
6.2 The Transaction File

The transaction file is established by means of the CLEAR command (see section 3.6.1). The filename of the transaction file is TRANS, and its extension is REC. A protection code is assigned to the file so that information can only be appended onto the end of the file. A malicious user cannot delete or supersede the transaction file. Student performance data is appended onto the end of the transaction file in dump mode which means that the data are in the same format as when they were in core. See Appendix E for details on the data format.

6.3 The Master Record File

Information in the transaction file is transferred to the master record file by the UPDATE command (see section 3.6.2). The filename of the master record file is MASTER, and its extension is REC. A protection code, which prevents users other than the instructor from gaining access to the file, is assigned automatically. Information from the transaction file is appended onto the master file with its format unchanged. Learn lesson reports will only include information from the master record file, no information from the transaction file will be included. The master record file may be transferred to DEC tape or ordinary magnetic tape for storage. This is an appropriate procedure to follow when disk space is limited.

6.4 Storage of Answer Strings

In addition to the student performance data that were described above, the actual answer strings which are typed by the students may also be stored. Whether or not the answer strings are stored is controlled by means of the STORE statement (see section 5.2.1.4.5). During lesson execution the strings are written into a temporary file in the student's disk area. At the completion of lesson execution, the Lesson Interpreter attempts to transfer the answer strings from the temporary file to an answer file that has been established under the instructor's identical project-programmer numbers using the CLEAR command (see section 3.6.1). The filename of the answer file is the same as the filename of the lesson, and its extension is ANS. A protection code is assigned so that information can only be appended to the end of the file. If the answer file cannot be found in the instructor's disk area, the temporary file in the student's disk area is preserved, and this file can be used as a source of data for answer reports.
6.5 Learn Reports

Learn reports may be printed on an instructor's terminal or on a line printer by using the REPORT command (see section 3.6.3). There are basically two kinds of reports - answer reports and reports based upon information in the master record file. These two kinds of reports are described below.

6.5.1 Answer Reports

In order for an answer report to be generated, the answer file for the lesson must be stored in the user's disk area. An answer report is obtained by including a /A switch in the REPORT command as shown in the following example.

REPORT LESN2/A

In the report, answers are grouped according to frame number. In addition to the answer string, the frame number, student programmer number, answer label, and right-wrong-neutral indicator for the answer are printed. An example of a line from an answer report is shown below.

0250 064304 1+ 75 NEWTONS

6.5.2 Reports Based Upon Master File Information

In order to obtain reports based upon student performance data, the master record file must be in the user's disk area. Reports may be classified in two different ways - as lesson reports or student reports and as summary reports or detailed reports. The four possible combinations are listed below.

1. Summary lesson reports
2. Summary student reports
3. Detailed lesson reports
4. Detailed student reports

Summary reports do not contain information on individual frames in a lesson, whereas detailed reports contain all the information that is included in a summary report plus information on each frame in a lesson. Summary reports may be printed on either a user's terminal or a line printer, but detailed reports can only be printed on a line printer. The Report Generator generates a detailed report if the device name LPT is specified in a REPORT command; otherwise, a summary report is generated.
In lesson reports, the information for all students that worked on a lesson is pooled, but in student reports, the information for each student is printed separately. The generation of student reports, especially detailed student reports, may require a relatively large amount of CPU time and a large number of pages of line printer paper. The Report Generator generates a lesson report when a lesson name is included in a REPORT command unless the /S switch is used.

Four examples of REPORT commands are shown below.

(a) REPORT LESN1
(b) REPORT 64000-64050
(c) REPORT LPT:LESN1
(d) REPORT LPT:LESN1/S

Example (a) generates a summary lesson report; example (b) generates a summary student report; example (c) generates a detailed lesson report; and example (d) generates a detailed student report.
This chapter describes the basic features that are available to a lesson author when a lesson is executed in test mode. When a lesson is executed in test mode, the normal sequence of lesson execution can be altered. This enables lesson authors to skip portions of a lesson that have already been tested, and it allows them to repeat a frame several times in order to try out different possible student responses. Storage of student performance data and answer strings is suppressed in order to prevent record files from being cluttered with useless data. Finally, the L: statement (see section 5.2.3.12) terminates a lesson, but it does not initiate the log-out process as it does during normal lesson execution.

7.1 Entering Test Mode

There are two ways to execute a lesson in test mode. One way is to use the TEST command (see section 3.4.2), and the other way is to exit from the Lesson Editor by means of the ET edit command (see section 4.4). Following either of these commands, the message

```
START AT FRAME *
```

is printed on the user's terminal. The user types the number of the frame at which he wishes to begin testing the lesson. Users should remember that skipping frames can result in unanticipated settings of Learn switches (see section 5.2.1.4). For example, if the user starts at frame 80, the EXTRA switch will be on even if frame 10 contains the statement EXTRA OFF>.

7.2 Test Mode Commands

There are three commands that may be used by lesson authors while they are executing a lesson in test mode. These commands may be typed following the asterisk that indicates that the Lesson Interpreter is ready to accept an answer.

7.2.1 The :n Command

Lesson authors may branch to any frame in a lesson by typing

```
:n
```
where $n$ is the frame number. For example, to branch to frame 150 the following command could be used.

>:150

If frame $n$ does not exist, the following message is printed on the user's terminal.

FRAME NOT FOUND. TRY AGAIN. *

The user may type another frame number following the asterisk.

7.2.2 The :E Command

If a lesson author discovers an error in a lesson during testing, he can enter the Lesson Editor to correct the error by typing :E. The Lesson Editor prints a greater than sign (>) to indicate that it is ready to accept a command. At this point the user may type any of the edit commands that are described in Chapter 4.

7.2.3 The :: Command

The user may terminate lesson testing at any time by typing a double colon (::). Learn responds by printing ENTER COMMAND.

7.3 An Example

The following example illustrates the use of the test mode commands.

```
ENTER COMMAND
*TEST ROCKS

TOTAL ERRORS DETECTED: 0

START AT FRAME *50

FM. 50

IGNEOUS ROCKS ARE FORMED FROM COOLING MAGMA. THE SLOWER THE MAGMA COOLS THE SMALLER THE SIZE OF THE CRYSTALS FORMED.

WRONG!
```

FM. 60

THE FACT THAT \( \uparrow 0 \)

FM. 50

IGNEOUS ROCKS ARE FORMED FROM COOLING MAGMA. THE SLOWER THE MAGMA COOLS THE LARGER THE SIZE OF THE CRYSTALS FORMED.

FINE!

FM. 60

THE FACT \( \uparrow 0 \)

FM. 120

APANITIC ROCKS FORM FROM MAGMA WHICH COOLS RAPIDLY. APHANITIC ROCKS ARE SAID TO BE INTRUSIVE.

>RI20,3,AP,APH

>ET

TOTAL ERRORS DETECTED: 0

START AT FRAME \( \uparrow 120 \)

FM. 120

APHANITIC ROCKS FORM FROM MAGMA WHICH COOLS RAPIDLY. APHANITIC ROCKS ARE SAID TO BE INTRUSIVE.

FINE!

7-3
PHANERITIC ROCKS FORM FROM MAGMA WHICH COOLS SLOWLY. PHANERITIC ROCKS ARE SAID TO BE...
CHAPTER 8

THE CALCULATION MODE

The calculation mode allows users of Learn to use the computer as a powerful calculator. Calculations are performed by means of the C-frame statements that were described in Chapter 5. This chapter describes some of the details of the calculation mode.

8.1 Entering the Calculation Mode

Learn users may enter the calculation mode by typing the ESC (or altmode) key. This may be done during a lesson following an asterisk which indicates that the Lesson Interpreter is ready to accept an answer, or it may be done at Learn command level following the message ENTER COMMAND. After the user types the ESC key, the following message is printed.

CALCULATION MODE

* 

At this point the user may begin performing his calculations. After all calculations have been completed, the user types the ESC key again to get out of calculation mode.

Access to the calculation mode during lesson execution may be restricted by means of the CALC statement (see section 5.2.1.4.1).

8.2 Calculation Mode Statements

The C-frame statements that are listed below may be used in the calculation mode.

1. ARRAY (see section 5.3.3.2)
2. assignment (see section 5.3.4)
3. INPUT (see section 5.3.5.1)
4. TYPE (see section 5.3.6.1)
5. ATYPE (see section 5.3.6.2)

None of the other C-frame statements are legal in the calculation mode.

When the calculation mode is entered from Learn command level, the CORE statement may be used to control the amount of core storage that is available. The form of the CORE statement is
CORE \( n \)

where \( n \) is a positive integer that is equal to the number of words of core that will be available. For example,

CORE 2000

would make 2000 words of core storage available to the user. If a CORE statement is not used, Learn automatically makes 128 words of core available. This applies only when the calculation mode is entered from Learn command level. The amount of calculation mode storage that is available during a lesson is controlled by the RESERVE statement (see section 5.3.10). The CORE statement may not be used during the execution of a lesson.

8.3 Some Restrictions

The following restrictions are in effect in the calculation mode:

1. No line numbers may be used (see section 5.3.1.1).

2. A maximum of 16 regular variable names (see section 5.3.2.3) or 8 array variable names (see section 5.3.2.4) may be used.

3. Special functions (see section 5.3.2.8) may not be used.

8.4 Some Example Calculations

```
ENTER COMMAND
* 5
CALCULATION MODE
*TYPE 24*15 ↓ 360
*TYPE EXP((3/4)*NLOG(16)) ↓ 8
8[I=1,10]TYPE I[5.0],SQRT(I)[9.3] ↓
   1 1.000
   2 1.414
   3 1.732
   4 2.000
   5 2.236
   6 2.449
   7 2.646
   8 2.828
   9 3.000
  10 3.162
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
* ARRAY AM(11) ↓
* AM(1) = 500.00 ↓
* [I=2,11] AM(I) = AM(I-1) + AM(I-1) * 0.05 ↓
* ATYPE AM(11)[8.2] ↓
  500.00
  525.00
  551.25
  578.81
  607.75
  638.14
  670.05
  703.55
  738.73
  775.66
  814.45
* ARRAY BX(5) ↓
* [I=1,5] INPUT BX(I) ↓
  2, 4, 6, 8, 10 ↓
* [I=1,5] M = M + BX(I) / 5 ↓
* ATYPE M ↓
  6
* $$
ENTER COMMAND
* BYE ↓

8-3

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The slide projector interface may be used to control either a Kodak Ektographic B or a Kodak Carousel AV 900 projector. In either case, the interface is connected to the output of a DTC acoustical data set in parallel with the student’s terminal, and it is also connected to the projector through its remote control receptacle. All characters that are sent to the terminal are monitored by the interface, but only four characters, none of which affect the terminal, are used to control the slide projector. A list of the seven bit codes of these characters is given below.

- `<02>` advance one slide
- `<03>` back up one slide
- `<04>` turn off the bulb
- `<06>` turn on the bulb

The projector slide tray must be set to zero at the beginning of a lesson, and the switch on the back of the projector must be set to the FAN position.

A circuit diagram of the projector interface is shown on the following pages. The signal from the acoustical coupler first passes through one of the two level converters that are shown on page A-2. A different level converter is needed for the teletype because the acoustical coupler is loaded down by the teletype circuitry. Each seven bit character code that is transmitted by the computer is preceded by a start pulse which sets the J-K flip-flop that is shown on page A-3. This in turn starts the clock. The clock pulses activate the shift registers allowing the seven bit code to be shifted into the registers. The clock pulses are counted by the decade counter, and after nine pulses, a reset pulse is sent to the J-K flip-flop to stop the clock. At this point, the three high order bits are analyzed by the NOR gates, and if all three high order bits are zero, the decoder for the four low order bits is enabled. The output from the decoder is sent to the relay circuits that are shown on page A-4. When the voltage at point E goes to zero, it closes a relay which turns the projector bulb on. The normally closed relay holds the other relay on until the voltage goes to zero at point D. Thus once the bulb is turned on, it stays on until the code `<04>` is sent to the interface. The NOR gates and the transistors on page A-4 form two one-shots which lengthen the low voltage output pulses from the decoder. This is required for reliable operation of the projector.

*The details of the design and the construction of the prototype were carried out by Mr. Darwin Parrish.*

---

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
LEVEL CONVERTER USED WITH ASR 33 TELETYPewriter

LEVEL CONVERTER USED WITH TEKTRONIX 4010 TERMINAL

A-2
enable decoder only if the 3 high order bits are zeroes

DECODING CIRCUIT

A-3
RELAY CIRCUITS

A-4

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
APPENDIX B

PHONETIC TRANSFORMATIONS

Learn phonetically processes lesson and student answers according to the procedure described below.

1. Examine the first character.
2. Go to 4.
3. Examine the next character.
4. If the character is an H, delete it and go to 3.
5. If the character is an I or an A, change it to an E and go to 12.
6. If the character is a P, V, or N, change it to E' and go to 12.
7. If the character is a G, change it to J and go to 12.
8. If the character is a C, S, or T, change it to K and go to 12.
9. If the character is an X, change it to Z and go to 12.
10. If the character is a carriage return, stop.
11. If the character is not a letter (A-Z), go to 3.
12. Examine the next character.
13. If the character is an A, E, H, I, O, U, W, or Y, delete it and go to 12.
14. If the character is the same as the last character, delete it and go to 12.
15. If the character is not an N, go to 18.
16. If the last character was a P, delete the N and go to 12.
17. Go to 12.
18. If the character is not a K, go to 21.
19. If the last character was a C, delete the K and go to 12.
20. Go to 12.
21. If the character is not a G, go to 24.
22. If the last character was an I, E, or N, delete the G and go to 12.
23. Change the G to an F, and go to 12.
24. If the character is a P, V, or J, change it to F and go to 12.
25. If the character is not an S, go to 29.
26. If the next character is a carriage return, delete the S and stop.
27. If the next character is not a letter (A-Z), delete the S and go to 3.
28. Change the S to K and go to 12.
29. If the character is a C, T, or X, change it to K and go to 12.
30. If the character is a carriage return, stop.
31. If the character is not a letter (A-Z), go to 3.

B-1
32. Go to 12.
Information files are used by instructors to pass messages or other information to their students. An information file is prepared in the same way as an ordinary lesson. Usually an information file has only a single Q-frame with a TEXT statement that contains the information. Learn prints the information on the user's terminal immediately after the greeting that is printed when the user begins running the Learn program (see section 1.3). An example information file is shown below.

```
10 TYPE-Q
TEXT
LESSON 4 MUST BE COMPLETED BY 3 PM ON JUNE 25.
```

C-1
A SUMMARY OF FILENAMES, EXTENSIONS, AND PROTECTION CODES

The table below contains a summary of filenames, extensions, and protection codes of Learn disk files.

<table>
<thead>
<tr>
<th>Type of File</th>
<th>Standard Filename</th>
<th>Standard Extension</th>
<th>Standard* Protection Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>unprocessed lesson</td>
<td>none**</td>
<td>UNP</td>
<td>&lt;177&gt;</td>
</tr>
<tr>
<td>processed lesson</td>
<td>same as unprocessed lesson</td>
<td>PRS</td>
<td>&lt;155&gt;</td>
</tr>
<tr>
<td>transaction file</td>
<td>TRANS</td>
<td>REC</td>
<td>&lt;447&gt;</td>
</tr>
<tr>
<td>master record file</td>
<td>MASTER</td>
<td>REC</td>
<td>&lt;177&gt;</td>
</tr>
<tr>
<td>answer file</td>
<td>same as lesson file</td>
<td>ANS</td>
<td>&lt;447&gt;</td>
</tr>
<tr>
<td>break*** file</td>
<td>same as lesson file</td>
<td>BRK</td>
<td>&lt;077&gt;</td>
</tr>
<tr>
<td>information file</td>
<td>INFO</td>
<td>PRS</td>
<td>&lt;155&gt;</td>
</tr>
<tr>
<td>help lesson</td>
<td>HELP</td>
<td>PRS</td>
<td>&lt;155&gt;</td>
</tr>
</tbody>
</table>

* See the DEC System 10 Users Handbook for a discussion of protection codes.
** Lesson names are assigned by lesson authors.
*** Break files contain information for resuming the lesson following a break (see section 5.5).
APPENDIX E

LEARN RECORD FILE FORMATS

Two words of data are stored for each frame in a lesson that is executed. The data format for Q-frames is as follows:

1st word - bits 0-10: frame number
   bits 11-17: letter indicating the frame type
   bits 18-35: time frame was executed (seconds)

2nd word - bits 0-6: answer label
   bits 7-13: right-wrong-neutral indicator
   bits 14-23: answering time in seconds
   bit 35 is set if the calculation mode was used

For other frame types, the 1st word is the same as for Q-frames, and the 2nd word is zero.

One line of ASCII information is written into an answer file for each answer that is typed by the student. The data format is as follows:

Characters 1-4: frame number
Characters 5-10: student's programmer number
Character 11: answer label
Character 12: right-wrong-neutral indicator
Rest of line: student's answer

Each line is terminated by a carriage return character only, a line feed is not present.
Device numbers are used by the Lesson Interpreter in executing a PAGE or P: statement. Each kind of terminal requires a different set of control characters to erase its screen, and the Lesson Interpreter uses the value of the special variable $D$ (see section 5.3.2.5) to determine which characters to send. A list of device numbers that have been assigned is given below. Additional numbers may be assigned in the future.

0 - Teletype
1 - Tektronix 4010
2 - Tektronix 4002

The value of the special variable $D$ should be set equal to the proper device number for the type of terminal that the student is using. This can be done by using a Q-frame that is similar to the one shown below.

```
5 TYPE-Q
TEXT
WHICH TERMINAL ARE YOU USING?
1) TELETYP
2) TEK 4010
3) TEK 4002

ANSWER
1 Ø 1
2 Ø 2
3 Ø 3
?

ACTIONS
1 2 3
C:$D=$A - 1
@ R:PLEASE TYPE 1, 2, OR 3.
```
This appendix contains a list of the values of the seven bit codes of the characters that may be used in the Learn Instructional Language.

<table>
<thead>
<tr>
<th>Character</th>
<th>Decimal Value</th>
<th>Character</th>
<th>Decimal Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>space</td>
<td>32</td>
<td>@</td>
<td>64</td>
</tr>
<tr>
<td>/</td>
<td>33</td>
<td>A</td>
<td>65</td>
</tr>
<tr>
<td>&quot;</td>
<td>34</td>
<td>B</td>
<td>66</td>
</tr>
<tr>
<td>#</td>
<td>35</td>
<td>C</td>
<td>67</td>
</tr>
<tr>
<td>$</td>
<td>36</td>
<td>D</td>
<td>68</td>
</tr>
<tr>
<td>%</td>
<td>37</td>
<td>E</td>
<td>69</td>
</tr>
<tr>
<td>&amp;</td>
<td>38</td>
<td>F</td>
<td>70</td>
</tr>
<tr>
<td>'</td>
<td>39</td>
<td>G</td>
<td>71</td>
</tr>
<tr>
<td>(</td>
<td>40</td>
<td>H</td>
<td>72</td>
</tr>
<tr>
<td>)</td>
<td>41</td>
<td>I</td>
<td>73</td>
</tr>
<tr>
<td>*</td>
<td>42</td>
<td>J</td>
<td>74</td>
</tr>
<tr>
<td>+</td>
<td>43</td>
<td>K</td>
<td>75</td>
</tr>
<tr>
<td>.</td>
<td>44</td>
<td>L</td>
<td>76</td>
</tr>
<tr>
<td>-</td>
<td>45</td>
<td>M</td>
<td>77</td>
</tr>
<tr>
<td>,</td>
<td>46</td>
<td>N</td>
<td>78</td>
</tr>
<tr>
<td>/</td>
<td>47</td>
<td>O</td>
<td>79</td>
</tr>
<tr>
<td>0</td>
<td>48</td>
<td>P</td>
<td>80</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
<td>Q</td>
<td>81</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>R</td>
<td>82</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
<td>S</td>
<td>83</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
<td>T</td>
<td>84</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
<td>U</td>
<td>85</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
<td>V</td>
<td>86</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
<td>W</td>
<td>87</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
<td>X</td>
<td>88</td>
</tr>
<tr>
<td>9</td>
<td>57</td>
<td>Y</td>
<td>89</td>
</tr>
<tr>
<td>;</td>
<td>58</td>
<td>Z</td>
<td>90</td>
</tr>
<tr>
<td>:</td>
<td>59</td>
<td>[</td>
<td>91</td>
</tr>
<tr>
<td>&lt;</td>
<td>60</td>
<td>\</td>
<td>92</td>
</tr>
<tr>
<td>=</td>
<td>61</td>
<td>]</td>
<td>93</td>
</tr>
</tbody>
</table>
Note: Lower case letters may be used in TEXT statements. The codes of lower case letters can be found by adding 64 to the value of the corresponding upper case letter.
# INDEX

## A
- abbreviated commands, 3-1
- A command, 4-2
- action labels, 5-14
- ACTION section, 5-2, 5-10
- ACTION statement, 5-10
- alphabetic labels, 5-7
- answer comparison switches, 5-9
- answer files, 2-3, 3-2, 6-2
- ANSWER section, 5-2, 5-6
- ANSWER statement, 5-6
- anticipated answers, 5-7
- ARRAY statement, 5-29, 8-1
- array variables, 5-24
- assignment statement, 5-29, 8-1
- A: statement, 5-11
- ATIME function, 5-27
- ATYPE statement, 5-33, 8-1

## D
- D command, 4-3
- DELETE command, 3-12
- DELIM statement, 5-1
- device numbers, 5-4, 5-13, 5-25
- F: command, 5-1
- disk, 1-2

## E
- E command, 4-7
- EDIT command, 3-3, 4-1
- EP command, 4-7
- error limits, 5-8, 5-25
- E: statement, 5-14
- ET command, 4-7
- EXTRA statement, 5-9

## B
- B-frames, 5-1, 5-39
- BOTH statement, 5-6, 5-17
- branching, 1-3
- B: statement, 5-12, 5-34
- BYE command, 3-27

## C
- CALC statement, 5-5, 8-1
- calculation mode, 1-3, 8-1
- CALL statement, 5-35
- C command, 4-3
- C-frames, 5-1, 5-21
- character constants, 5-23
- CLEAR command, 2-3, 3-20, 6-2
- constants, 5-23
- COPY command, 3-10
- CORE statement, 8-1
- CRT displays, 1-2, 5-4, 5-13
- C: statement, 5-13

## F
- filename, D-1
- FILES command, 3-16
- FILL statement, 5-31
- format specification, 5-3, 5-32
- frame number, 2-1, 4-2, 5-1, 6-1
- frame number increment, 3-2, 3-4, 4-2, 4-7
- frame type, 2-1, 5-1
- F: statement, 5-10
- functions
  - mathematical, 5-26
  - special, 5-27

## G
- GET command, 2-4, 3-7
- GOTO statement, 5-34
- G: statement, 5-12, 5-34

---

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
H
HELP command, 3-1
HELP lesson, 3-1
high segment, 1-5
hint label, 5-6, 5-16
HINT statement, 5-5

I
I command, 4-3
IF statement, 5-34
index variables, 5-22
information files, C-1
INPUT statement, 5-30, 8-1
interaction fields, 5-21

K
K command, 4-4
KJOB command, 3-26

L
Lesson Editor, 1-5, 4-1
lesson filename, 2-1, 3-2, 3-5, D-1
Lesson Interpreter, 1-5
Lesson Processor, 1-5
LIL, 1-3, 5-1
line number, 2-1, 4-1, 5-21
LIST command, 3-17
log-in, 1-2
log-out, 1-4
L: statement, 5-14, 7-1

M
MACRO-10, 1-4
master record file, 2-4, 3-22, 6-2
MATCH statement, 5-8
mathematical expressions, 5-22
M command, 4-7

monitor, 1-1
MONITOR command, 3-1, 3-26
N
N command, 4-5
NEUT function, 5-27
numeric label, 5-7

O
operators, 5-25
ORDER statement, 5-10

P
PAGE statement, 5-4, F-1
password, 1-2
PATH function, 5-28
peripheral devices, 1-2
phonetic comparison, B-1
PHONIC statement, 5-10
PROCESS command, 2-2, 3-5
processed lesson, 1-5
project-programmer number, 1-2, 1-4, 3-7
protection codes, 2-1, 3-2, 3-5, 3-21, 6-2, D-1
P: statement, 5-13, F-1

Q
Q-frames, 5-1
Q: statement, 5-12

R
random number, 5-25
R command, 4-5
record keeping, 1-3, 6-1
RECORD statement, 5-6, 6-1
re-entrant, 1-4

INDEX-2

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
regular variables, 5-24
relational symbols, 5-35
R-frames, 5-1, 5-38
REM statement, 5-17, 5-36
RENAME command, 3-18
REPORT command, 2-5, 3-23
Report Generator, 1-5
reports, 1-3, 2-4, 6-3
RESERVE statement, 5-36, 8-2
RETURN statement, 5-35
REVIEW statement, 5-5
RIGHT function, 5-27
right-wrong-neutral indicator, 5-7, 6-1
R: statement, 5-11
SCALAR statement, 5-29
SEEN function, 5-27
slide projector, 1-2, A-1
slide projector interface, A-1
SLIDE statement, 5-3
special variables, 5-24
S: statement, 5-13
statement fields, 5-22
STORE statement, 2-4, 5-6, 6-2
TAG function, 5-28
T command, 4-6
TEST command, 3-8
test mode, 1-4, 7-1
test mode commands, 7-1
TEXT section, 5-2
TEXT section switches, 5-4
TEXT statement, 5-2
timesharing, 1-1
time up label, 5-16
transaction file, 2-3, 3-21, 6-1, 6-2
TYPE statement, 5-31, 8-1

unanticipated answers, 5-6, 5-8
unconditional actions, 5-17
unprocessed lesson, 1-5
UPDATE command, 2-4, 3-22
WRITE command, 2-1, 3-2, 4-1
WRONG function, 5-27
zero labels, 5-8

INDEX-3
A LISTING OF LEARN SYSTEM PROGRAMS

The rest of this dissertation consists of a cross-reference listing of all the programs that comprise the Learn system. Information on the availability of Learn can be obtained by writing to the address shown below.

Learn
The Computer Center
Western Michigan University
Kalamazoo, Michigan 49001
LEARN IS AN INTEGRATED SYSTEM OF PROGRAMS FOR COMPUTER-ASSISTED INSTRUCTION. LEARN PROGRAMS ARE WRITTEN IN MACRO-10, AND THEY MAY BE RUN WITH EITHER A KS10 OR KS10 GPU RUNNING UNDER THE CONTROL OF A 906 MONITOR. THE LEARN SYSTEM WAS DEVELOPED BY GARRETT A. VANDER LUGT AS PART OF HIS DOCTORAL RESEARCH AT WESTERN MICHIGAN UNIVERSITY.

OTHER DOCUMENTATION FOR THE LEARN SYSTEM INCLUDES THE "LEARN LESSON AUTHOR'S GUIDE" AND THE LEARN HELP LESSON (HELP,UNP),

LEARN FILES


LEARN.SHR IS COMPOSED OF THE FOLLOWING FILES:

LEARN.MAC
INTERP.MAC
CALC.MAC
CPROS.MAC
FUNCS.MAC
ACTION.MAC
AUX1.MAC
LODL.MAC (DUMMY LOW SEGMENT)

LEDIT.SHR IS COMPOSED OF THE FOLLOWING FILES:

LEDIT.MAC
PERL.MAC
LPRINT.MAC
WRTEQ.MAC
AUX2.MAC
LODLLOW.MAC (DUMMY LOW SEGMENT)

LRPRT.SHR IS COMPOSED OF THE FOLLOWING FILES:

LRPRT.MAC
RPRT.MAC
LPRPT.MAC
RPRINT.MAC
AUX3.MAC
RPTLOW.MAC (DUMMY LOW SEGMENT)

LPROS.SHR IS COMPOSED OF THE FOLLOWING FILES:

LPROS.MAC
PROS.MAC
CPRAM.MAC
AUX4.MAC
PRSLW.MAC (DUMMY LOW SEGMENT)

THE FILES FOR EACH HIGH SEGMENT SHOULD BE LOADED IN THE ORDER SHOWN ABOVE AND SAVED USING THE SAVE MONITOR COMMAND.

OTHER FILES IN THE LEARN SYSTEM ARE LEARN.DOC AND HELP,UNP.
THE PROCESSED HELP LESSON (HELP,PRS) SHOULD BE PLACED ON
DEVICE SYS ALONG WITH THE FOUR REENTRANT HIGH SEGMENTS.

NEITHER CARREY A. VANDERLUGT NOR WESTERN MICHIGAN UNIVERSITY IS UNDER ANY OBLIGATION TO MAINTAIN THE LEARN SYSTEM. FURTHERMORE,
NONE OF THE FEATURES DESCRIBED IN ANY LEARN SYSTEM DOCUMENTATION IS GUARANTEED.

SEE INDIVIDUAL FILES FOR ADDITIONAL DOCUMENTATION.

NO ERRORS DETECTED

PROGRAM BREAK IS 88888

3K CORE USED
TITLE LEARN

COMMENT

WRITTEN BY GARRET A. VANDER LUST COMPLETED 2-MAY-74

PART OF LEARN, SHR, VERSION 3

*****************************************************************

SPECIAL NOTE:

LEARN, MAC CONTAINS THE STARTING ADDRESS FOR THE ENTIRE LEARN SYSTEM.
AFTER A USER TYPES THE "R LEARN" MONITOR COMMAND, EXECUTION BEGINS
AT LOCATION "START" IN SECTION 6 OF THIS PROGRAM.

THE LEARN HIGH SEGMENT IS COMPOSED OF THE FOLLOWING SUBPROGRAMS:

1. LEARN, MAC
2. INTERP, MAC
3. CALC, MAC
4. CPROSS, MAC
5. FUNCS, MAC
6. ACTION, MAC
7. AUX1, MAC
8. LOW1, MAC (DUMMY LOW SEGMENT).

THESE PROGRAMS SHOULD BE LOADED IN THE ORDER IN WHICH THEY ARE LISTED
ABOVE. AFTER LOADING, A REENTRANT, BINARY CORE IMAGE OF THE LEARN
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SSAVE MONITOR COMMAND.

THE LEARN HIGH SEGMENT EXECUTES THE FOLLOWING LEARN SYSTEM COMMANDS:

1. BYE
2. "ESCH" - USED TO ENTER THE CALCULATION MODE
3. GET
4. HELP
5. KBID
6. MONITOR
7. TEST

*****************************************************************

LEARN, MAC INITIALIZES THE LEARN SYSTEM AND THEN TYPES A GREETING
WHICH IS APPROPRIATE FOR THE TIME OF DAY. IT ALSO INTERPRETS THE
FIRST WORD IN LEARN SYSTEM COMMANDS AND CALLS THE PROPER SUBPROGRAM
TO EXECUTE THEM. IF A COMMAND CANNOT BE EXECUTED BY THE LEARN HIGH
SEGMENT, THEN LEARN, MAC PLACES A COMMAND INDEX IN LOCATION STORE AND
LEARN MACRO 47(284)-7 88105 2-AUG-74 PAGE 1-1

54 REPLACES ITSELF WITH THE HIGH SEGMENT WHICH CAN EXECUTE THE COMMAND.
55 WHEN LEARN.MAC IS ENTERED FROM ANOTHER HIGH SEGMENT, IT RETRIEVES A
56 COMMAND INDEX FROM LOCATION STORE AND CALLS THE PROPER SUBPROGRAM TO
57 EXECUTE THE COMMAND INDICATED BY THE COMMAND INDEX. LEARN.MAC ALSO
58 DOES LOOKUPS FOR LESSONS SPECIFIED IN LEARN "GET" COMMANDS.

CONTENTS OF LEARN.MAC

SECTION A. PERFORMS THE GETSEG UUQ.

SECTION B. CONTAINS THE ENTRY POINT TO THE LEARN HIGH SEGMENT.

SECTION C. CONTAINS THE STARTING ADDRESS FOR THE LEARN SYSTEM.

SECTION D. TYPES AN APPROPRIATE GREETING.

SECTION E. INPUTS THE FIRST WORD IN A LEARN COMMAND.

SECTION F. INTERPRETS THE FIRST WORD IN LEARN COMMAND.

SECTION G. EXECUTES A "MONITOR" COMMAND.

SECTION H. PUTS THE USER IN THE CALCULATION MODE.

SECTION I. SETS UP GETSEG ARGUMENTS FOR THE LEDIT HIGH SEGMENT.

SECTION J. SETS UP GETSEG ARGUMENTS FOR THE LRPT HIGH SEGMENT.

SECTION K. SETS UP GETSEG ARGUMENTS FOR THE LPROS HIGH SEGMENT.

SECTION L. CONTINUES THE EXECUTION OF A TEST COMMAND.

SECTION M. INPUTS THE LESSON NAME IN A "GET" COMMAND.

SECTION N. INPUTS PROJ-PROG NUMBERS IN A "GET" COMMAND.

SECTION O. DOES A LOOKUP FOR THE LESSON SPECIFIED IN A "GET"

SECTION P. EXECUTES A "BYE" COMMAND.

SECTION Q. EXECUTES A "KJOB" COMMAND.

ENTRY POINTS TO LEARN.MAC

BYE - FROM INTERP.MAC (SECTION WH) IN ORDER TO EXECUTE AN "L", LOCATED IN SECTION P

GND - FROM INTERP.MAC (SECTION WH) AT THE CONCLUSION OF A

LESSON, LOCATED IN SECTION E
GETSG = FROM INTERP.MAC (SECTION C0) IN RESPONSE TO A TEST MODE
IE COMMAND, LOCATED IN SECTION A

RETURN = FROM ANOTHER HIGH SEGMENT, LOCATED IN SECTION B

START = FROM MONITOR LEVEL, LOCATED IN SECTION C

EXTERNAL SUBRoutines CALLED BY LEARN.MAC

CALCM = PERFORMS CALCULATION MODE FUNCTIONS, LOCATED IN CALC.MAC
(SECTION A)

EDLIN = INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE
END OF A LINE IS REACHED, LOCATED IN AUX1.MAC (SECTION A)

INTERP = INTERPRETS A PROCESSED LESSON FILE, LOCATED IN INTERP.MAC
(SECTION A)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOH1.MAC)

ADDR = ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO
VARIABLES USED IN THE CALCULATION MODE

ARGINT = ARGUMENTS FOR GETSEG UDO (6 WORDS)

DATA = STARTING ADDRESS FOR STORAGE OF STUDENT VARIABLES,
LESSON VARIABLES, AND USER PERFORMANCE DATA

IBUF = INPUT BUFFER HEADER (3 WORDS)

INAM = LOOKUP ARGUMENTS (4 WORDS)

INBUF = DISK INPUT BUFFER (31 WORDS)

INTBLK = CONTROL C INTERRUPT BLOCK (5 WORDS)

INTLOC = CONTROL C INTERRUPT ROUTINE (14 WORDS)

JOBNO = USER'S JOB NUMBER

LIGN = LESSON LOOKUP ARGUMENTS

OBUF = OUTPUT BUFFER HEADER (3 WORDS)

ONAM = ENTER ARGUMENTS (4 WORDS)

OUTBUF = DISK OUTPUT BUFFER (13 WORDS)

PPN = USER'S PROG-PROG NUMBERS
RSVNUM - NUMBER OF WORDS RESERVED FOR USER CALC MODE DATA
STORE - COMMAND INDEX IS SAVED HERE DURING A GETSEG
SYMIDX - CALC MODE SYMBOL TABLE INDEX
TRMBLK - TRMOP, ARGUMENTS (3 WORDS)

DESCRIPTION OF FLAGS STORED IN AC=9

BIT 0 TEST LESSON
BIT 1 INFORMATION LESSON

X

0000081  MISEG

000001  INTERM_GETSEG, BYE, CMD
000002  EXTERN ARGMT, CALCH, EDLIN, IBUF, INAH, INBUF, INTBLK, INTREP, INTLOC, JOBNO,
000003  EXTERN OBUF, ONAH, OUTBUF, PRT, RSVNUM, STORE, TRMBLK, LKSAV, DATA_ADDR, SYMIDX

0000000  FLAGS=8
000001  A=1
000002  B=2
000003  C=3
000004  D=4
000005  E=5
000006  F=6
000007  G=7
000017  G=17

SECTION A. GET THE PROPER HIGH SEGMENT TO EXECUTE THE COMMAND
WHICH WAS TYPED BY THE USER. EXECUTION RESUMES IN LEDIT, MAC, ILPROS, MAC, OR LRPRT, MAC

NOTE: NO DATA OR INSTRUCTIONS SHOULD BE PLACED AHEAD OF "GETSEG". SINCE THE RETURN ADDRESS OF THE GETSEG WGO MUST BE THE SAME IN ALL HIGH SEGMENTS.

0000000  GETSEG A;
000001  GETSEG A;
000002  MALT

SECTION B. RETRIEVE THE COMMAND INDEX WHICH WAS PLACED IN LOCATION
"STORE" BY THE PREVIOUS HIGH SEGMENT AND JUMP TO THE PROPER ROUTINE
1 TO EXECUTE THE COMMAND.
SECTION C. INITIALIZE THE PROGRAM, SAVE ARGUMENTS FOR GETSEG UUDO

WHICH HAVE BEEN PLACED IN ACCUMULATORS B, 7, AND 11 BY THE MONITOR,

OPEN CHANNELS 1 AND 2 AND ESTABLISH BUFFERS, TRANSFER A ROUTINE

IF HANDLING CONTROL 0'S TO THE LOW SEGMENT.

POINT1 POINT 7,0,35

START: SETEO F,ADDR ...STARTING ADDRESS FOR LEARN

RESTART: RSET 0,32 ...RESTART HERE AFTER CONTROL C

TYPE "MON" TO EXIT FROM LEARN

/3
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>564</td>
<td>002449</td>
<td>JST CHD</td>
</tr>
<tr>
<td>566</td>
<td>002441</td>
<td>JST CALL 4, A</td>
</tr>
<tr>
<td>568</td>
<td>002447</td>
<td>ENDNUM</td>
</tr>
<tr>
<td>584</td>
<td>002445</td>
<td>INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>586</td>
<td>002451</td>
<td>GETHE:</td>
</tr>
<tr>
<td>588</td>
<td>002453</td>
<td>MOVE B, INAM+3</td>
</tr>
<tr>
<td>590</td>
<td>002455</td>
<td>JST PPINS</td>
</tr>
<tr>
<td>592</td>
<td>002457</td>
<td>YES, GO TO PPINS</td>
</tr>
<tr>
<td>594</td>
<td>002459</td>
<td>JST CHKPPIN+1</td>
</tr>
<tr>
<td>596</td>
<td>002461</td>
<td>JST GETL</td>
</tr>
</tbody>
</table>

**SECTION D. PERFORM LOOKUP FOR LESSON FILE.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>601</td>
<td>002463</td>
<td>GETLES:</td>
</tr>
<tr>
<td>603</td>
<td>002465</td>
<td>GETINF:</td>
</tr>
<tr>
<td>605</td>
<td>002467</td>
<td>MOVE B, INAM+3</td>
</tr>
<tr>
<td>607</td>
<td>002469</td>
<td>GETL1:</td>
</tr>
<tr>
<td>609</td>
<td>002471</td>
<td>MOVE B, XHDLN (INAM, LKAVY)</td>
</tr>
<tr>
<td>611</td>
<td>002473</td>
<td>BIT LXKAVY+3</td>
</tr>
<tr>
<td>613</td>
<td>002475</td>
<td>MOVEX LUXKAVY+3</td>
</tr>
<tr>
<td>615</td>
<td>002477</td>
<td>MOVE B, INAM+1</td>
</tr>
<tr>
<td>617</td>
<td>002479</td>
<td>MOVE B, INAM</td>
</tr>
<tr>
<td>619</td>
<td>002481</td>
<td>MOVE B, INAM</td>
</tr>
<tr>
<td>621</td>
<td>002483</td>
<td>MOVE B, INAM+1</td>
</tr>
<tr>
<td>623</td>
<td>002485</td>
<td>MOVE B, INAM+3</td>
</tr>
<tr>
<td>625</td>
<td>002487</td>
<td>MOVE B, INAM+3</td>
</tr>
<tr>
<td>627</td>
<td>002489</td>
<td>MOVE B, INAM+3</td>
</tr>
<tr>
<td>629</td>
<td>002491</td>
<td>MOVE B, INAM+3</td>
</tr>
</tbody>
</table>

**SECTION E. EXECUTE THE BYE COMMAND BY RUNNING THE LOGOUT PROGRAM.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>631</td>
<td>002501</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>633</td>
<td>002503</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>635</td>
<td>002505</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>637</td>
<td>002507</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>639</td>
<td>002509</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>641</td>
<td>002511</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>643</td>
<td>002513</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>645</td>
<td>002515</td>
<td>JST CALL 3, ASCI</td>
</tr>
<tr>
<td>647</td>
<td>002517</td>
<td>JST CALL 3, ASCI</td>
</tr>
</tbody>
</table>

**NOTE:** The code appears to be a mix of assembler or汇编代码, which is commonly used in computer programming. The code is designed to perform specific tasks, likely related to file management, input processing, and file lookup operations. The usage of assembler or汇编语言 suggests that it is being used in a low-level programming environment, possibly for system-level tasks such as operating systems or high-performance applications. The specific tasks include handling character inputs, moving data between registers and memory, and executing system-specific commands or programs.
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>637</td>
<td>000514' 000 00 0 0 000000</td>
<td>BYE1</td>
</tr>
<tr>
<td>638</td>
<td>000517' 051 11 0 0 000000</td>
<td>TDCALL 11,6</td>
</tr>
<tr>
<td>639</td>
<td>000520' 201 01 0 0 000511</td>
<td>MOVE1 A, RUNARG</td>
</tr>
<tr>
<td>640</td>
<td>000521 047 81 0 0 000039</td>
<td>RUN A;</td>
</tr>
<tr>
<td>641</td>
<td>000522 204 04 0 0 000008</td>
<td>HALT</td>
</tr>
<tr>
<td>642</td>
<td></td>
<td></td>
</tr>
<tr>
<td>643</td>
<td></td>
<td></td>
</tr>
<tr>
<td>644</td>
<td></td>
<td></td>
</tr>
<tr>
<td>645</td>
<td></td>
<td></td>
</tr>
<tr>
<td>646</td>
<td>000523 63 71 63 00 00 00</td>
<td>RARG1</td>
</tr>
<tr>
<td>647</td>
<td>000524 53 52 57 42 00 00</td>
<td>SIXBIT /KJOB/</td>
</tr>
<tr>
<td>648</td>
<td>000525 000 00 00 00 000000</td>
<td></td>
</tr>
<tr>
<td>649</td>
<td>000526 000 00 00 00 000000</td>
<td></td>
</tr>
<tr>
<td>650</td>
<td>000527 000 00 00 00 000000</td>
<td></td>
</tr>
<tr>
<td>651</td>
<td>000530 000 00 00 00 000000</td>
<td></td>
</tr>
<tr>
<td>652</td>
<td>000531 251 04 0 0 000001</td>
<td>TDCALL 4, A</td>
</tr>
<tr>
<td>653</td>
<td>000532 306 01 0 0 000040</td>
<td>KJOB1</td>
</tr>
<tr>
<td>654</td>
<td>000533 254 02 0 0 000031</td>
<td>JUSTR 1, 2</td>
</tr>
<tr>
<td>655</td>
<td>000534 306 01 0 0 000037</td>
<td>CAIN A, X</td>
</tr>
<tr>
<td>656</td>
<td>000535 254 02 0 0 000041</td>
<td>JUSTR KJ</td>
</tr>
<tr>
<td>657</td>
<td>000536 302 01 0 0 000015</td>
<td>CAIN A, 15</td>
</tr>
<tr>
<td>658</td>
<td>000537 254 00 0 0 000205</td>
<td>JUSTR CMDERR 1NO, GO TYPE ERROR MESSAGE (SECTION E)</td>
</tr>
<tr>
<td>659</td>
<td>000540 031 04 0 0 000001</td>
<td>TDCALL 4, A</td>
</tr>
<tr>
<td>660</td>
<td>000541 201 01 0 0 000023</td>
<td>KJ1</td>
</tr>
<tr>
<td>661</td>
<td>000542 047 81 0 0 000039</td>
<td>RUN A;</td>
</tr>
<tr>
<td>662</td>
<td>000543 204 04 0 0 000008</td>
<td>HALT</td>
</tr>
<tr>
<td>663</td>
<td></td>
<td></td>
</tr>
<tr>
<td>664</td>
<td></td>
<td></td>
</tr>
<tr>
<td>665</td>
<td></td>
<td></td>
</tr>
<tr>
<td>666</td>
<td></td>
<td></td>
</tr>
<tr>
<td>667</td>
<td>0000241</td>
<td>NOSYM</td>
</tr>
<tr>
<td>668</td>
<td></td>
<td>END START</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 00070E

3K CORE USED
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>164#</td>
<td>205</td>
<td>256</td>
<td>257</td>
<td>258</td>
<td>259</td>
<td>260</td>
<td>261</td>
<td>262</td>
<td>263</td>
</tr>
<tr>
<td></td>
<td>264</td>
<td>265</td>
<td>266</td>
<td>267</td>
<td>268</td>
<td>269</td>
<td>270</td>
<td>271</td>
<td>272</td>
<td>273</td>
</tr>
<tr>
<td></td>
<td>274</td>
<td>275</td>
<td>276</td>
<td>277</td>
<td>278</td>
<td>279</td>
<td>280</td>
<td>281</td>
<td>282</td>
<td>283</td>
</tr>
<tr>
<td></td>
<td>284</td>
<td>285</td>
<td>286</td>
<td>287</td>
<td>288</td>
<td>289</td>
<td>290</td>
<td>291</td>
<td>292</td>
<td>293</td>
</tr>
<tr>
<td></td>
<td>294</td>
<td>295</td>
<td>296</td>
<td>297</td>
<td>298</td>
<td>299</td>
<td>300</td>
<td>301</td>
<td>302</td>
<td>303</td>
</tr>
<tr>
<td></td>
<td>304</td>
<td>305</td>
<td>306</td>
<td>307</td>
<td>308</td>
<td>309</td>
<td>310</td>
<td>311</td>
<td>312</td>
<td>313</td>
</tr>
<tr>
<td></td>
<td>314</td>
<td>315</td>
<td>316</td>
<td>317</td>
<td>318</td>
<td>319</td>
<td>320</td>
<td>321</td>
<td>322</td>
<td>323</td>
</tr>
<tr>
<td></td>
<td>324</td>
<td>325</td>
<td>326</td>
<td>327</td>
<td>328</td>
<td>329</td>
<td>330</td>
<td>331</td>
<td>332</td>
<td>333</td>
</tr>
<tr>
<td></td>
<td>334</td>
<td>335</td>
<td>336</td>
<td>337</td>
<td>338</td>
<td>339</td>
<td>340</td>
<td>341</td>
<td>342</td>
<td>343</td>
</tr>
<tr>
<td></td>
<td>344</td>
<td>345</td>
<td>346</td>
<td>347</td>
<td>348</td>
<td>349</td>
<td>350</td>
<td>351</td>
<td>352</td>
<td>353</td>
</tr>
<tr>
<td></td>
<td>354</td>
<td>355</td>
<td>356</td>
<td>357</td>
<td>358</td>
<td>359</td>
<td>360</td>
<td>361</td>
<td>362</td>
<td>363</td>
</tr>
<tr>
<td></td>
<td>364</td>
<td>365</td>
<td>366</td>
<td>367</td>
<td>368</td>
<td>369</td>
<td>370</td>
<td>371</td>
<td>372</td>
<td>373</td>
</tr>
<tr>
<td></td>
<td>374</td>
<td>375</td>
<td>376</td>
<td>377</td>
<td>378</td>
<td>379</td>
<td>380</td>
<td>381</td>
<td>382</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>385</td>
<td>386</td>
<td>387</td>
<td>388</td>
<td>389</td>
<td>390</td>
<td>391</td>
<td>392</td>
<td>393</td>
</tr>
<tr>
<td></td>
<td>394</td>
<td>395</td>
<td>396</td>
<td>397</td>
<td>398</td>
<td>399</td>
<td>400</td>
<td>401</td>
<td>402</td>
<td>403</td>
</tr>
<tr>
<td></td>
<td>404</td>
<td>405</td>
<td>406</td>
<td>407</td>
<td>408</td>
<td>409</td>
<td>410</td>
<td>411</td>
<td>412</td>
<td>413</td>
</tr>
<tr>
<td></td>
<td>414</td>
<td>415</td>
<td>416</td>
<td>417</td>
<td>418</td>
<td>419</td>
<td>420</td>
<td>421</td>
<td>422</td>
<td>423</td>
</tr>
<tr>
<td></td>
<td>424</td>
<td>425</td>
<td>426</td>
<td>427</td>
<td>428</td>
<td>429</td>
<td>430</td>
<td>431</td>
<td>432</td>
<td>433</td>
</tr>
<tr>
<td></td>
<td>434</td>
<td>435</td>
<td>436</td>
<td>437</td>
<td>438</td>
<td>439</td>
<td>440</td>
<td>441</td>
<td>442</td>
<td>443</td>
</tr>
<tr>
<td></td>
<td>444</td>
<td>445</td>
<td>446</td>
<td>447</td>
<td>448</td>
<td>449</td>
<td>450</td>
<td>451</td>
<td>452</td>
<td>453</td>
</tr>
<tr>
<td></td>
<td>454</td>
<td>455</td>
<td>456</td>
<td>457</td>
<td>458</td>
<td>459</td>
<td>460</td>
<td>461</td>
<td>462</td>
<td>463</td>
</tr>
<tr>
<td></td>
<td>464</td>
<td>465</td>
<td>466</td>
<td>467</td>
<td>468</td>
<td>469</td>
<td>470</td>
<td>471</td>
<td>472</td>
<td>473</td>
</tr>
<tr>
<td></td>
<td>474</td>
<td>475</td>
<td>476</td>
<td>477</td>
<td>478</td>
<td>479</td>
<td>480</td>
<td>481</td>
<td>482</td>
<td>483</td>
</tr>
<tr>
<td></td>
<td>484</td>
<td>485</td>
<td>486</td>
<td>487</td>
<td>488</td>
<td>489</td>
<td>490</td>
<td>491</td>
<td>492</td>
<td>493</td>
</tr>
</tbody>
</table>

|     | 494  | 495  | 496  | 497  | 498  | 499  | 500  | 501  | 502  | 503  |
|     | 504  | 505  | 506  | 507  | 508  | 509  | 510  | 511  | 512  | 513  |
|     | 514  | 515  | 516  | 517  | 518  | 519  | 520  | 521  | 522  | 523  |
|     | 524  | 525  | 526  | 527  | 528  | 529  | 530  | 531  | 532  | 533  |
|     | 534  | 535  | 536  | 537  | 538  | 539  | 540  | 541  | 542  | 543  |
|     | 544  | 545  | 546  | 547  | 548  | 549  | 550  | 551  | 552  | 553  |
|     | 554  | 555  | 556  | 557  | 558  | 559  | 560  | 561  | 562  | 563  |
|     | 564  | 565  | 566  | 567  | 568  | 569  | 570  | 571  | 572  | 573  |
|     | 574  | 575  | 576  | 577  | 578  | 579  | 580  | 581  | 582  | 583  |
|     | 584  | 585  | 586  | 587  | 588  | 589  | 590  | 591  | 592  | 593  |

<table>
<thead>
<tr>
<th>ADDR</th>
<th>318#</th>
<th>319#</th>
<th>320#</th>
<th>321#</th>
<th>322#</th>
<th>323#</th>
<th>324#</th>
<th>325#</th>
<th>326#</th>
<th>327#</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFN</td>
<td>313#</td>
<td>314#</td>
<td>315#</td>
<td>316#</td>
<td>317#</td>
<td>318#</td>
<td>319#</td>
<td>320#</td>
<td>321#</td>
<td>322#</td>
</tr>
<tr>
<td>ARGMT</td>
<td>323#</td>
<td>324#</td>
<td>325#</td>
<td>326#</td>
<td>327#</td>
<td>328#</td>
<td>329#</td>
<td>330#</td>
<td>331#</td>
<td>332#</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>ILLPN</th>
<th>560</th>
<th>577</th>
<th>583</th>
<th>593</th>
<th>524</th>
<th>525</th>
<th>561</th>
<th>594</th>
<th>597</th>
<th>606</th>
<th>608</th>
<th>609</th>
<th>611</th>
</tr>
</thead>
<tbody>
<tr>
<td>INAM</td>
<td>162#</td>
<td>327</td>
<td>432</td>
<td>434</td>
<td>524</td>
<td>525</td>
<td>561</td>
<td>594</td>
<td>597</td>
<td>606</td>
<td>608</td>
<td>609</td>
<td>611</td>
</tr>
<tr>
<td>INBUF</td>
<td>182#</td>
<td>258</td>
<td>263</td>
<td>264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTEK</td>
<td>193#</td>
<td>293</td>
<td>298</td>
<td>300</td>
<td>303</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INLOC</td>
<td>185#</td>
<td>285</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTRN</td>
<td>265#</td>
<td>302</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOBNO</td>
<td>182#</td>
<td>277</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KJ</td>
<td>687</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KJOB</td>
<td>427</td>
<td>634#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEAVE</td>
<td>461#</td>
<td>467</td>
<td>469</td>
<td>471</td>
<td>473</td>
<td>479</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEAVE2</td>
<td>482#</td>
<td>486</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEAVE3</td>
<td>505#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIST</td>
<td>615#</td>
<td>619</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LKSAV</td>
<td>167#</td>
<td>430</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MESS</td>
<td>317</td>
<td>320</td>
<td>323#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOMIT</td>
<td>428</td>
<td>447#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HORN</td>
<td>312#</td>
<td>321#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAMER</td>
<td>535#</td>
<td>539#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>527#</td>
<td>538#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NONAME</td>
<td>523#</td>
<td>544#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOTFND</td>
<td>612#</td>
<td>614#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTFND2</td>
<td>615#</td>
<td>623#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OBUF</td>
<td>238</td>
<td>267</td>
<td>261</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DNM</td>
<td>193#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPI</td>
<td>231#</td>
<td>252</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPI2</td>
<td>236#</td>
<td>254</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTBUF</td>
<td>236#</td>
<td>256</td>
<td>265</td>
<td>266</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POINT</td>
<td>267#</td>
<td>280</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPN1</td>
<td>571#</td>
<td>593#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPN2</td>
<td>574#</td>
<td>592#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRCCS</td>
<td>504#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>435#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAC</td>
<td>444#</td>
<td>472#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RENAME</td>
<td>428#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPORT</td>
<td>419#</td>
<td>481#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESTRT</td>
<td>247#</td>
<td>294</td>
<td>298</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSVNUM</td>
<td>103#</td>
<td>219</td>
<td>269</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUNARG</td>
<td>639#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SITNT</td>
<td>261#</td>
<td>306#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>START</td>
<td>236#</td>
<td>688</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STORE</td>
<td>263#</td>
<td>272</td>
<td>279</td>
<td>280</td>
<td>335</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYMIX</td>
<td>216</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST</td>
<td>416#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST2</td>
<td>220#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRNBLK</td>
<td>143#</td>
<td>272</td>
<td>273</td>
<td>275</td>
<td>280</td>
<td>335</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATE</td>
<td>423#</td>
<td>487#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE</td>
<td>433#</td>
<td>468#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TITLE INTERP

COMMENT X

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 3-MAY-74

PART OF LEARN.SHR, VERSION 3

INTERP.MAC INTERPRETS PROCESSED LESSON FILES.  IN PARTICULAR;
IT INTERPRETS Q-FRAMES, R-FRAMES, AND B-FRAMES.  C-FRAMES ARE
INTERPRETED BY CALC.MAC.  INTERP.MAC ALSO STORES STUDENT PERFOR-
MANCE DATA.

NOTE:  SEE PROS.MAC AND CFRAM.MAC FOR A LIST OF THE STATEMENT
CODES WHICH MAY APPEAR IN A PROCESSED LESSON FILE.

CONTENTS OF INTERP.MAC

SECTION A.  Initializes the lesson interpreter.

SECTION B.  Interprets the first line in a frame.

SECTION C.  Interprets the first character in a Q-frame statement.

SECTION D.  Sets CALC, HINT, BOTH, STORE, REVIEW, AND RECORD-FLAGS.

SECTION E.  Types the text in a text statement.

SECTION F.  Interprets SLIDE statements and PAGES CRT TERMINALS.

SECTION G.  Accepts student answers.

SECTION H.  Processes HINT REQUESTS, REVIEW REQUESTS, AND
CALC MODE REQUESTS.

SECTION I.  Interprets ANSWER TAGS, ANSWER EVALUATION MODE
STATEMENTS, AND RIGHT-WRONG-NEUTRAL INDICATORS.

SECTION J.  Converts student answers to floating point nos.

SECTION K.  Interprets CALC statements after "P" tags.

SECTION L.  Sets extra, order, and phonic flags.

SECTION M.  Interprets MATCH statements.

SECTION N.  Interprets DELIM statements.

SECTION O.  Sets up list of standard word delimiters.

SECTION P.  Selects proper answer evaluation mode.
### SECTION Q
Extra on and order off answer evaluation routine.

### SECTION R
Extra off and order off answer evaluation routine.

### SECTION S
Extra on and order on answer evaluation routine.

### SECTION T
Extra on and order off answer evaluation routine.

### SECTION U
Extra on and order on answer comparison subroutine.

### SECTION V
Extra on and order off answer comparison subroutine.

### SECTION W
Phonetically processes student answers.

### SECTION X
Subroutines used in answer evaluation.

### SECTION Y
Compares numerical lesson and student answers.

### SECTION Z
Finds the actions for a particular answer.

### SECTION AA
Subroutines for skipping text statements.

### SECTION BB
Stores student answers.

### SECTION CC
Stores student performance data.

### SECTION DD
Interprets B-frames.

### SECTION EE
Interprets R-frames.

### SECTION FF
Executes review commands.

### SECTION GG
Executes test commands.

### SECTION HH
Terminates lesson and stores lesson data.

### ENTRY POINTS TO INTERP, MAC

- ANSW = FROM ACTION, MAC (SECTION D), located in SECTION Q
- EOL = FROM CALC, MAC (SECTION A) and ACTION, MAC (SECTIONS H & J), located in SECTION HH
- FSTLIN = FROM CALC, MAC (SECTIONS A, P & D) and ACTION, MAC (SECTIONS J & K), located in SECTION B
- GETACT = FROM ACTION, MAC (SECTIONS A, C, E, G & L), located in SECTION Z
- INTERP = FROM LEARN, MAC (SECTION D), located in SECTION A
LOGOUT - FROM ACTION.MAC (SECTION I), LOCATED IN SECTION HH

PAG - FROM ACTION.MAC (SECTION L), LOCATED IN SECTION F

QFRAME - FROM ACTION.MAC (SECTION F), LOCATED IN SECTION C

SKPTXT - FROM CALC.MAC (SECTION P) AND ACTION.MAC (SECTION J), LOCATED IN SECTION AA

SLDRTN - FROM ACTION.MAC (SECTION G), LOCATED IN SECTION F

UNCAL - FROM ACTION.MAC (SECTIONS A, C, G & L), LOCATED IN SECTION 2

EXTERNAL SUBROUTINES USED BY INTERP.MAC

ACMD - TYPES THE CORRECT ANSWERS, LOCATED IN ACTION.MAC (SECTION E)

BCHD - EXECUTES BRANCHING (BI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION K)

BCHD2 - EXECUTES BRANCHING, LOCATED IN ACTION.MAC (SECTION K)

BYE - RUNS THE LOGOUT PROGRAM, LOCATED IN LEARN.MAC (SECTION P)

CALC - ENTRY POINT TO THE CALCULATION MODE, LOCATED IN CALC.MAC (SECTION A)

CHD - EXECUTES CALC (CI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION C)

CFRAME - EXECUTES C-FRAME STATEMENTS, LOCATED IN CALC.MAC (SECTION A)

ECHD - EXECUTES EXIT (EI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION H)

EXPOR - EXPANDS THE LOW SEGMENT IF NECESSARY, LOCATED IN AUX1.MAC (SECTION H)

FCMD - EXECUTES FEEDBACK (FI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION A)

FINT - CONVERTS THE NO. IN AC-A TO AN INTEGER, LOCATED IN FUNS.MAC (SECTION B)

GCHD - EXECUTES SUBROUTINE CALLS (G1), LOCATED IN ACTION.MAC (SECTION J)

GETSG - EXECUTES THE GETSEG UUD, LOCATED IN LEARN.MAC (SECTION A)
160 110 = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1.MAC (SECTION E)

161 1102 = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1.MAC (SECTION E)

162 INFLP = "INPUTS" A FLOATING POINT NO., INTO AC-B, LOCATED IN AUX1.MAC (SECTION F)

166 IOT = OUTPUTS THE INTEGER IN AC-A TO THE USER'S TERMINAL, LOCATED IN AUX1.MAC (SECTION D)

170 LCMD = EXECUTES LOGOUT (LI) STATEMENTS, LOCATED IN AUX1.MAC (SECTION I)

174 QCMD = EXECUTES PAGE (P1) STATEMENTS, LOCATED IN AUX1.MAC (SECTION L)

180 POLISH = EVALUATES POLISH EXPRESSIONS, LOCATED IN CALC.MAC (SECTION D)

185 QCMD = EXECUTES REPEAT QUESTION (Q1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION F)

190 RCMD = EXECUTES REPEAT (R1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION D)

200 RETSUB = RETURNS FROM A SUBROUTINE, LOCATED IN ACTION.MAC (SECTION J)

205 SCMD = EXECUTES SLIDE (S1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION G)

210 UCMD = EXECUTES CALC-(G1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION C)

220 UCMD = EXECUTES FEEDBACK-(FI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION A)

230 UPCMD = EXECUTES PAGE-(PI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION L)

240 USCMD = EXECUTES SLIDE-(S1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION G)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOH1.MAC)
ADDR = ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO VARIABLES USED IN STUDENT CALCULATION MODE

ANSTIM = INITIAL ANSWERING TIME, ACCUMULATED ANSWERING TIME, AND WAIT TIME (3 WORDS)

ARGMT = GETSET ARGUMENTS (6 WORDS)

BLOCK = DISK BLOCK NO., BUFFER BYTE POINTER, AND BUFFER BYTE COUNT AT BEGINNING OF LAST ANSWER STATEMENT (3 WORDS)

COMNUM = RANDOM NO. USED TO SELECT RANDOM COMMENTS.

COUNT = NO. OF WORDS IN AN ANSWER THAT MUST BE PRESENT FOR A MATCH AND NO. OF WORDS ACTUALLY PRESENT (2 WORDS)

DATA = STARTING ADDRESS FOR STORAGE OF STUDENT VARIABLES, LESSON VARIABLES, AND STUDENT PERFORMANCE DATA

DELM = LIST OF WORD DELIMITERS (17 WORDS)

DEVICE = DEVICE NO. ASSOCIATED WITH USER'S TERMINAL

FRNUM = CURRENT FRAME NO.

FRMTOT = TOTAL NO. OF FRAMES EXECUTED

FRMTYP = FRAME TYPE OF CURRENT FRAME

HINTC = NO. OF HINTS REQUESTED AND NO. OF HINT-TAGS ENCOUNTERED IN THE ACTION SECTION (2 WORDS)

IBUF = DISK INPUT BUFFER HEADER (3 WORDS)

IDX = INDEX IS SAVED HERE WHEN AC-F IS USED BY ANOTHER SUBPROGRAM

INDEX = INDEX TO THE 1ST WORD CONTAINING STUDENT PERFORMANCE DATA

INAM = DISK LOOKUP ARGUMENTS (4 WORDS)

LANS = LESSON ANSWER BUFFER (16 WORDS)

LESHAD = LESSON ANSWER WORD HOLDER (4 WORDS)

NNUM = NUMBER OF NEUTRAL ANSWERS

OFLAGS = INTERPRETER STORES FLAGS HERE

ONAM = DISK ENTER ARGUMENTS (4 WORDS)

PC2 = SUBROUTINES STORE RETURN ADDRESS HERE
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>266</td>
<td>POLLST = POLISH STRING PUSH DOWN LIST (64 WORDS)</td>
</tr>
<tr>
<td>267</td>
<td>PPN = USER'S PROJECT-PROGRAMMER NO.</td>
</tr>
<tr>
<td>268</td>
<td>RANNUM = INTEGER RANDOM NO. USED BY RANDOM FUNCTION</td>
</tr>
<tr>
<td>271</td>
<td>REVIDX = REVIEW TABLE INDEX</td>
</tr>
<tr>
<td>272</td>
<td>REVNUM = REVIEW TABLE (16 WORDS)</td>
</tr>
<tr>
<td>275</td>
<td>RGANS = CORRECT ANSWER BUFFER (16 WORDS)</td>
</tr>
<tr>
<td>277</td>
<td>RNUM = NO. OF CORRECT ANSWERS</td>
</tr>
<tr>
<td>280</td>
<td>RPTCNT = NO. OF TIMES FRAME HAS BEEN REPEATED AND NO. OF TIMES A '2' TAG HAS BEEN ENCOUNTERED IN THE ACTION SECTION OF A Q-FRAME (2 WORDS)</td>
</tr>
<tr>
<td>281</td>
<td>RSVNUM = TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE</td>
</tr>
<tr>
<td>287</td>
<td>RUTH = INITIAL CPU TIME AT BEGINNING OF LESSON</td>
</tr>
<tr>
<td>289</td>
<td>SANS = STUDENT ANSWER BUFFER (16 WORDS)</td>
</tr>
<tr>
<td>290</td>
<td>SANS2 = PHONETICALLY PROCESSED STUDENT ANSWER BUFFER (16 WORDS)</td>
</tr>
<tr>
<td>293</td>
<td>SAVPNT = USED BY ANSWER EVALUATION ROUTINES TO STORE BYTE POINTERS (2 WORDS)</td>
</tr>
<tr>
<td>296</td>
<td>SAVPOS = INFORMATION NEEDED TO CONTINUE THE LESSON AFTER A REVIEW SEQUENCE - DISK BLOCK NO., BUFFER BYTE POINTER TO BEGINNING OF FRAME, BUFFER BYTE COUNT, FRAME NO. (4 WORDS)</td>
</tr>
<tr>
<td>299</td>
<td>SLDMEM = CURRENT SLIDE NO.</td>
</tr>
<tr>
<td>301</td>
<td>SNANS = NUMERIC VALUE OF LAST STUDENT ANSWER</td>
</tr>
<tr>
<td>304</td>
<td>STDATE = DATE LESSON EXECUTION WAS STARTED</td>
</tr>
<tr>
<td>306</td>
<td>STWROD = USED BY STUDENT ANSWER ROUTINE TO STORE A WORD FROM A STUDENTS ANSWER (4 WORDS)</td>
</tr>
<tr>
<td>308</td>
<td>STORE (*OSKBLK) = CURRENT DISK BLOCK NO.</td>
</tr>
<tr>
<td>311</td>
<td>SUBPDP = PUSH DOWN LIST FOR STORING INFORMATION NEEDED TO RETURN FROM A SUBROUTINE - 1ST WORD CONTAINS PUSH DOWN POINTER (10 WORDS)</td>
</tr>
<tr>
<td>315</td>
<td>SQED = INFORMATION NEEDED TO REPEAT QUESTION (0) - DISK BLOCK NO., BUFFER BYTE POINTER, AND BUFFER-BYTE-COUNT (3 WORDS)</td>
</tr>
<tr>
<td>318</td>
<td>SYMHID = INDEX TO CALC MODE SYMBOL TABLE</td>
</tr>
</tbody>
</table>
TAG = TAG OF THE LAST ANSWER

TAGCNT = NO. OF ANSWER TAGS THAT HAVE BEEN STORED IN THE TAG BUFFER AND THE NO. OF TIMES THE TAG OF THE CURRENT ANSWER HAS BEEN ENCOUNTERED IN THE ACTION SECTION (2 WORDS)

TAGPNT = BYTE POINTER TO TAG BUFFER

TAGREC = TAG BUFFER (2 WORDS)

TAGREP = NO. OF TIMES ANSWER WITH THIS TAG HAS BEEN REPEATED BY THE STUDENT

TIME = TIME OF DAY LESSON WAS STARTED

TNUM = NO. OF TIME UPS

TOL = NUMERIC ANSWER ERROR TOLERANCE

TRMLK = TRHOP, ARGUMENTS (3 WORDS)

WNUM = NO. OF WRONG ANSWERS

DESCRIPTION OF FLAGS STORED IN AC=8

BIT 8 TEST LESSON
BIT 1 INFORMATION LESSON
BIT 2 NOT USED
BIT 3 CORRECT ANSWER
BIT 4 WRONG ANSWER
BIT 9 TEMPORARY ANSWER EVALUATION MODE
BIT 6 PHONIC ON
BIT 7 ORDER ON
BIT 8 EXTRA ON
BIT 9 TEMPORARY PHONIC ON
BIT 10 TEMPORARY ORDER ON
BIT 11 TEMPORARY EXTRA ON
BIT 12 ANSWER CONVERTED TO FLOATING POINT NUMBER
BIT 13 CORRECT ANSWER STORED
BIT 14 NUMERIC LESSON ANSWER
BIT 15 END OF LESSON ANSWER
BIT 16 END OF STUDENT ANSWER
BIT 17 ANSWER PHONETICALLY PROCESSED
BIT 18 NUMERIC STUDENT ANSWER
BIT 19 EXTRA WORD IN STUDENT ANSWER
BIT 20 NO MATCH
BIT 21 NOT USED
BIT 22 NOT USED
BIT 23 NOT USED
BIT 24 NOT USED
BIT 25 BULB ON
INTREP MACRO 47(264)=7 08186 2-AUG-74 PAGE 1-8

EXTERN FRMTOT,RNUM,MNUM,NNUM,TNUM,DATA,NUMTH,ANSTIM
EXTERN ID,IOT,SLDNUM,TAG,STORE,BLOCK,IBUF
EXTERN SANS,LANS,FRNUM,FRNTPY,PC2,PPN
EXTERN ACHD,BCMD,BCMD2,SCMD,FCMD,GCMD,GCMD,RCMD
EXTERN RESUB,SCMD,UCMD,UPCMD,OFLAGS,LCMD,USCMD,BYE,GETSG

000003 FLG#8
000001 A#1
000002 B#2
000003 C#3
000004 D#4
000005 E#5
000006 INDEX#6
000007 G#7
000010 H#10
000016 OFLAGS#16
000017 Q#17

POIN T1: POINT 10, DATA (INDEX) .23
POIN T2: POINT 7, SANS
POIN T3: POINT 3, POINT 7, LANS
POIN T4: POINT 11, DATA (INDEX) .10
POIN T5: POINT 7, DATA (INDEX) .17
POIN T6: POINT 7, DATA (INDEX) .16
POIN T7: POINT 7, DATA (INDEX) .13
POIN T8: POINT 6, SANS
POIN T9: POINT 6, LESWRD
POIN T10: POINT 7, SANS
POIN T11: POINT 10, REVNUM (D) .9
POIN T12: POINT 26, REVNUM (D) .35
POIN T13: POINT 7, RGTANS
TPOIN T: POINT 7, YAGREÇ
CRLF0: ASCII /
ASTRIS: ASCII /#/!
DYSCE5: DDS400000

SECTION A, INITIALIZE THE LESSON INTERPRETER AND LOOK FOR
BREAK FILE.

IDISK INPUT ROUTINE WHICH IS TRANSFERRED TO AC-11 THRU AC-15
RTUNI S0JUL 15, GETBUF IF BUFFER IS EMPTY, GO FILL IT (AUX1,MAC)
10DB A,14 IGET CHARACTER FROM BUFFER
SECTION E. TYPE OUT THE TEXT IN A TEXT STATEMENT.

TEXT:

JSP Q,11  INPUT CHARACTER
JUMPE A.ENDTXT IF CHARACTER IS <09, GO TO ENDTXT
TCALL A, I TYPE CHARACTER
JSTX Text 1GO INPUT NEXT CHARACTER
ENDTXT:  JSP Q,11  IS CHARACTER A RETURN?
JSTQFRAM1  YES, GO TO GFRAM (SECTION C)
JRT OFRAM3 ISE THE ANSWER FILE OPEN FLAG
JSTQFRAM1  YES, GO TO GFRAM (SECTION C)
JRT OFRAM3  ISE THE ANSWER FILE OPEN FLAG
SECTION F. INTERPRET A SLIDE STATEMENT AND PAGE CRT TERMINALS.

02371 265 17 0 00 00375  PAGEI JSP GPag  ICALL PACING ROUTINE
02374 254 00 0 00 02351  JRST GFRAMES  1GO TO GFRAMES (SECTION C)

IFE K116,PAGI MOVE A,DEVICE  IGET THE DEVICE NO.
JSP 0,FINT  ICONVERT TO FIXED POINT>
IFN K116,PAGI FIX A,DEVICE  IGET DEVICE NO.>
JUMP A,01  IRETURN IF ZERO (TTY)
MOVE C,1
MOVE D,TRBLK  IGET ADDRESS OF TRMLK, ARGUMENTS
TRMLK, D,  ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY
JRST +3  IHALT A SEC BEFORE CHECKING BUFFER AGAIN
JRST +3  1GO CHECK BUFFER AGAIN
MOVE B,2
SLEEP B,  IPAUSE BEFORE ERASING SCREEN

CALL A,5  IIBS DEVICE NO. 19
CALL A,6  IIBS DEVICE NO. 20 (TEK 4010)
CALL A,7  IIBS DEVICE NO. 21
JRST TK4002  IYES, GO TO TK4002 (TEK 4002)
TCCALL 3,FASC16 /

UNDEFINED DEVICE NUMBER

02413 251 03 0 00 02651  JRT (D)  RETURN.
02414 254 00 0 17 02000  JRST (D)  RETURN.
02415 201 01 0 00 02003  TK4015  MOVE A,33
02416 201 15 0 00 0001  TCCALL 16,A  IOUTPUT <33>
02417 201 01 0 00 0001  MOVE A,14
02418 201 15 0 00 0001  TCCALL 16,A  IOUTPUT <14>
02419 201 01 0 00 0001  MOVE A,14
02420 401 03 0 00 0000  WMPG D  cis\      ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY
02421 401 03 0 00 0000  JRST +3  IHALT A JIFFY BEFORE TRYING AGAIN
02422 401 03 0 00 0000  JRST +3  1GO CHECK BUFFER AGAIN
02423 401 03 0 00 0000  MOVE C,1
02424 401 03 0 00 0000  SLEEP C,  IWAIT A SEC
02425 051 15 0 00 0000  TCCALL 5,FASC2D  IRETURN
02426 051 15 0 00 0000  H4OVE A,30
02427 051 15 0 00 0000  TCCALL 16,A  IOUTPUT <30>
02428 051 15 0 00 0000  MOVE A,1
02429 051 15 0 00 0000  TCCALL 16,A  IOUTPUT <12>
02430 051 00 0 00 0045  JRST WTSEG  IGO HAIT A SEC.

OFFSTR 4
02431 00028 00003 00026  ONSTR 6
02432 00029 00026 00003  REVER 3
02433 00028 00002 00002  FWRDI 2
02434 025 17 0 00 08445  SLIDEI JSP 0,SLDRTN  ICALL SLIDE, INTERPRETER
02444 202 83 0 00 00235  JRST GFRAMES  1GO TO GFRAMES (SECTION C)

...
SLDTRI TDCALL 15,FRWD TYPE SLIDE PROJ, CONTROL CHARACTERS
TRMP, D) ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

SLDTRI TDCALL 15,REVR3 TYPE SLIDE PROJ, CONTROL CHARACTERS
TRMP, D) ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

SLDTRI TDCALL 15,ONSTR ELSE TURN IT OFF

SLDTRI TDCALL 15,FRWD TYPE SLIDE PROJ, CONTROL CHARACTERS
TRMP, D) ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

SLDTRI TDCALL 15,REVR3 TYPE SLIDE PROJ, CONTROL CHARACTERS
TRMP, D) ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

SLDTRI TDCALL 15,ONSTR ELSE TURN IT OFF
<table>
<thead>
<tr>
<th>Address</th>
<th>Assembly Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>849</td>
<td>000552</td>
<td>SLEEP C, IPause</td>
</tr>
<tr>
<td>850</td>
<td>000553</td>
<td>IJP Q, 11 INPUT CHARACTER</td>
</tr>
<tr>
<td>851</td>
<td>000554</td>
<td>CAIC A, 15 IS CHARACTER A RETURN?</td>
</tr>
<tr>
<td>852</td>
<td>000555</td>
<td>JAST SLOTM IND, GO TO SLOTM</td>
</tr>
<tr>
<td>853</td>
<td>000556</td>
<td>JAST &amp;PCC IRETURN</td>
</tr>
<tr>
<td>854</td>
<td>000557</td>
<td>SLOONI TCALL 15, ONSTR I TYPE SLIDE PROJ. CONTROL CHARACTERS</td>
</tr>
<tr>
<td>855</td>
<td>000558</td>
<td>SLEEP C, IPause</td>
</tr>
<tr>
<td>856</td>
<td>000559</td>
<td>TRO FLAGS, 2000 SET BULB ON</td>
</tr>
<tr>
<td>857</td>
<td>000562</td>
<td>JAST &amp;PCC IRETURN</td>
</tr>
<tr>
<td>858</td>
<td>000563</td>
<td>SLOOFFI TCALL 15, OFFSTR I TYPE SLIDE PROJ. CONTROL CHARACTERS</td>
</tr>
<tr>
<td>859</td>
<td>000564</td>
<td>SLEEP C, IPause</td>
</tr>
<tr>
<td>860</td>
<td>000565</td>
<td>T3E FLAGS, 2000 CLEAR BULB ON FLAG</td>
</tr>
<tr>
<td>861</td>
<td>000566</td>
<td>JAST &amp;PCC IRETURN</td>
</tr>
<tr>
<td>862</td>
<td>000567</td>
<td>SLO1M1 SUB1 A, 40 I SUBTRACT TO GET PROJECTION TIME</td>
</tr>
<tr>
<td>863</td>
<td>000568</td>
<td>IMULI A, $1000 I CONVERT SECONDS TO MILLISECONDS</td>
</tr>
<tr>
<td>864</td>
<td>000569</td>
<td>MSTIME B, I GET CURRENT TIME</td>
</tr>
<tr>
<td>865</td>
<td>000570</td>
<td>SLPI SLEEP C, I WAIT A SEC BEFORE CHECKING TIME</td>
</tr>
<tr>
<td>866</td>
<td>000571</td>
<td>MSTIME D, I GET NEXT CURRENT TIME</td>
</tr>
<tr>
<td>867</td>
<td>000572</td>
<td>SUB D, 8 I SUBTRACT TO GET ELAPSED TIME</td>
</tr>
<tr>
<td>868</td>
<td>000573</td>
<td>SKIPG D I SKIP IF THIS IS NOT THE NEXT DAY</td>
</tr>
<tr>
<td>869</td>
<td>000574</td>
<td>ADD D, DYESecs I ADD NO. OF MSEC'S IN A DAY</td>
</tr>
<tr>
<td>870</td>
<td>000575</td>
<td>CAGE D A I HAS PROJECTION TIME EXPIRED?</td>
</tr>
<tr>
<td>871</td>
<td>000576</td>
<td>JAST SLP I ND, GO WAIT A LITTLE LONGER</td>
</tr>
<tr>
<td>872</td>
<td>000577</td>
<td>JAST &amp;PCC IRETURN</td>
</tr>
</tbody>
</table>

SECTION 5: ACCEPT A STUDENT'S ANSWER,

Answer: MOVE B. DSKBLK

MOVE B. BLOCK ISAVE CURRENT DISK BLOCK IN BLOCK

MOVE B. BLOCK ISAVE BUFFER BYTE POINTER IN BLOCK

MOVE B. BLOCK ISAVE BUFFER BYTE COUNTER IN BLOCK

ANSWI T3E FLAGS, 1 I SET STUDENT ANSWER PHONETICALLY PROCESSED FLAG

SET NUMERIC ANSWER TOLERANCE TO ZERO

EVALUATION MODE, TEMP, EXTRA, TEMP, ORDER, AND TEMP, PHONIC

T3E FLAGS, 4000 I IS EXTRA SET?

T3E FLAGS, 4000 I YES, SET TEMP, EXTRA ON

T3E FLAGS, 2000 I IS ORDER SET?

T3E FLAGS, 2000 I YES, SET TEMP, ORDER ON

T3E FLAGS, 1000 I IS PHONIC SET?

T3E FLAGS, 1000 I YES, SET TEMP, PHONIC ON

T3E FLAGS, 1000 I CLEAR ANSWER CONVERSION FLAG

MOVE A. TRMBLK I PUT ADDRESS OF TRMOP, ARGUMENTS IN AQ=A

MOVE B. A I SKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

TRMOP, A I

SLEEP B I SLEEP A SEC BEFORE CHECKING BUFFER AGAIN

JAST +3 I GO CHECK BUFFER AGAIN

JAST +3 I JAST +3

TSCALL 14 I OVERRIDE USER'S CONTROL Q, IF ANY

JFCL
INTERP MACRO 47 (204) 7 0018 2-AUG-74 PAGE 1-22

1167 001195 254 00 00 0001377 JST MTEST YES, GO TO MTEST (SECTION M)
1168 001196 306 01 00 000020 CAIN A, 28 DELIM STATEMENTS
1169 001197 254 00 00 0001374 JST DELSET YES, GO TO DELSET (SECTION N)
1170 001198 305 01 00 000060 CAIN A, "0" CALC STATEMENT?
1171 001199 254 00 00 001275 JST CLOSTM YES, GO TO CLOSTM (SECTION K)
1172 001121 202 01 00 000134 CAIG A, "7"
1173 001122 302 01 00 000061 CAIGE A, "1" JST EVAL3 ISKIP, IF ALPHABETIC TAG
1174 001125 254 00 00 001171 TRF FLAGS, 480000 ISET NUMERIC ANSWER FLAG
1175 001126 600 00 00 480060 TRLN FLAGS, 40 TOSTUDENT ANSWER BEEN CONVERTED?
1176 001127 607 00 00 000040 JST CNVANS GO TO CNVANS (SECTION J)
1177 001128 621 00 00 001233 EVAL3 TLE FLAGS, 68000 ICLEAR CORRECT AND INCORRECT ANSWER FLAGS
1178 001129 265 17 00 000051 JSP Q, 11 INPUT RIGHT-WRONG-NEUTRAL INDICATOR
1179 001121 306 01 00 000053 CAIN A, "0" IIS THIS A CORRECT ANSWER?
1180 001121 661 00 00 400000 CAIN A, "1" YES, SET CORRECT ANSWER FLAG
1181 001121 306 00 00 000053 CAIN A, "7" IIS THIS AN INCORRECT ANSWER?
1182 001121 661 00 00 400000 CAIN A, "1" ISET INCORRECT ANSWER FLAG
1183 001121 202 02 00 000062 MOVE B, POINT3 IPUT BYTE POINTER TO LANS IN AC-B
1184 001121 765 17 00 000061 JSP Q, 11 INPUT CHARACTER
1185 001121 136 01 00 000060 JSP A, B IPUT CHARACTER IN LANS
1186 001121 302 01 00 000065 CAIG A, "15 IEND OF ANSWER?
1187 001121 254 00 00 001200 JST -, - INO, GO INPUT NEXT CHARACTER
1188 001121 607 00 00 004000 TLEN NTANS, 480000 TNO THIS A CORRECT ANSWER?
1189 001121 254 00 00 001200 JST EVAL4 INO, GO TO EVAL4
1190 001121 603 00 00 000060 TLEN NTANS, 20 IHAS CORRECT ANSWER BEEN STORED?
1191 001121 254 00 00 001200 TLEN NTANS, 20 YES, GO TO EVAL4
1192 001121 602 00 00 480000 TLEN NTANS, 480000 TNO ANSWER NUMERIC?
1193 001121 661 00 00 000060 TLENS TANS, 18 ISET STORED ANSWER NUMERIC FLAG
1194 001121 202 02 00 000062 MOVE B, (CNVANS, LANS, RCTANS) ISTORE ANSWER IN RCTANS
1195 001121 251 02 00 000062 BLT B, RCTANS+17 ISTORE ANSWER IN RCTANS
1196 001121 661 00 00 000060 TLO NTANS, 20 ISET CORRECT ANSWER STORED FLAG
1197 001121 200 01 00 000162 EVAL4 MOVE A, TAG IGET TAG IN AC-A
1198 001121 302 01 00 000077 CAIN A, "7" JIS TAG A "7"?
1199 001121 254 00 00 000072 JST COUTNT INO, GO TO COMP
1200 001121 350 00 00 000175 CAIN A, "17" IADD ONE TO COUNT?
1201 001121 202 01 00 000104 MOVE A, OFLAGS IPUT FLAGS IN AC-A
1202 001121 604 01 00 001200 TRNN A, 100 ISET, BOTH SET?
1203 001121 254 00 00 001200 JST - DD INO, SKIP
1204 001121 602 01 00 000060 CAIN A, "17" IYES, SKIP
1205 001121 602 01 00 000060 CAIN A, "17" IYES, ADD ONE TO COUNT.
1206 001121 254 00 00 001200 JST + 3 INO, SKIP
1207 001121 602 01 00 000060 CAIN A, 400 IIS BOTH ON?
1208 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.
1209 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.
1210 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.
1211 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.
1212 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.
1213 001121 254 00 00 001200 CAIN A, 400 IYES, ADD ONE TO HINT COUNT.

SECTION J. CONVERT STUDENT ANSWERS TO FLOATING POINT NUMBERS;

1214 001231 420 00 00 000001 CNVANS TLE FLAGS, 4 ICLEAR ZERO TAG FLAG
1215 001234 259 17 00 001235 JFOL 17, +1 ICLEAR CPU FLAGS
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1273</td>
<td>FSETI</td>
<td>TIE FLAGS, 10000 (\text{CLEAR TEMP, ANSWER EVALUATION MODE FLAG})</td>
</tr>
<tr>
<td>1274</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1275</td>
<td>MOVE B, A</td>
<td>SAVE THE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>1276</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1277</td>
<td>CAIN A, &quot;T&quot;</td>
<td>IS CHARACTER A (&quot;T&quot;)</td>
</tr>
<tr>
<td>1278</td>
<td>TIE FLAGS, 10000 (\text{SET TEMP, ANSWER EVALUATION MODE FLAG})</td>
<td></td>
</tr>
<tr>
<td>1279</td>
<td>JSP Q, 11</td>
<td>INPUT RETURN CHARACTER</td>
</tr>
<tr>
<td>1280</td>
<td>TIE FLAGS, 10000 (\text{IIS TEMP, MODE FLAG SET})</td>
<td></td>
</tr>
<tr>
<td>1281</td>
<td>JSP Q, 11</td>
<td>INPUT RETURN CHARACTER</td>
</tr>
<tr>
<td>1282</td>
<td>IYES, JUMP</td>
<td></td>
</tr>
<tr>
<td>1283</td>
<td>TIE FLAGS, 4400 (\text{CLEAR EXTRA ON FLAGS})</td>
<td></td>
</tr>
<tr>
<td>1284</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1285</td>
<td>TIE FLAGS, 4400 (\text{YES, SET EXTRA ON})</td>
<td></td>
</tr>
<tr>
<td>1286</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
<tr>
<td>1287</td>
<td>TIE FLAGS, 4400 (\text{CLEAR, EXTRA ON FLAG})</td>
<td></td>
</tr>
<tr>
<td>1288</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1289</td>
<td>TIE FLAGS, 4400 (\text{SET, EXTRA ON})</td>
<td></td>
</tr>
<tr>
<td>1290</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
<tr>
<td>1291</td>
<td>OSETI</td>
<td>TIE FLAGS, 10000 (\text{CLEAR TEMP, ANSWER EVALUATION MODE FLAG})</td>
</tr>
<tr>
<td>1292</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1293</td>
<td>MOVE B, A</td>
<td>SAVE THE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>1294</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1295</td>
<td>CAIN A, &quot;T&quot;</td>
<td>IS CHARACTER A (&quot;T&quot;)</td>
</tr>
<tr>
<td>1296</td>
<td>TIE FLAGS, 10000 (\text{SET TEMP, ANSWER EVALUATION MODE FLAG})</td>
<td></td>
</tr>
<tr>
<td>1297</td>
<td>JSP Q, 11</td>
<td>INPUT RETURN CHARACTER</td>
</tr>
<tr>
<td>1298</td>
<td>TIE FLAGS, 10000 (\text{IIS TEMP, MODE FLAG SET})</td>
<td></td>
</tr>
<tr>
<td>1299</td>
<td>IYES, JUMP</td>
<td></td>
</tr>
<tr>
<td>1300</td>
<td>TIE FLAGS, 2200 (\text{CLEAR ORDER ON FLAGS})</td>
<td></td>
</tr>
<tr>
<td>1301</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1302</td>
<td>TIE FLAGS, 2200 (\text{YES, SET ORDER ON})</td>
<td></td>
</tr>
<tr>
<td>1303</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
<tr>
<td>1304</td>
<td>TIE FLAGS, 2200 (\text{CLEAR ORDER ON FLAG})</td>
<td></td>
</tr>
<tr>
<td>1305</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1306</td>
<td>TIE FLAGS, 2200 (\text{SET, ORDER ON})</td>
<td></td>
</tr>
<tr>
<td>1307</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
<tr>
<td>1308</td>
<td>PHSSETI</td>
<td>TIE FLAGS, 10000 (\text{CLEAR TEMP, ANSWER EVALUATION MODE FLAG})</td>
</tr>
<tr>
<td>1309</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1310</td>
<td>MOVE B, A</td>
<td>SAVE THE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>1311</td>
<td>JSP Q, 11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1312</td>
<td>CAIN A, &quot;T&quot;</td>
<td>IS CHARACTER A (&quot;T&quot;)</td>
</tr>
<tr>
<td>1313</td>
<td>TIE FLAGS, 10000 (\text{SET TEMP, ANSWER EVALUATION MODE FLAG})</td>
<td></td>
</tr>
<tr>
<td>1314</td>
<td>JSP Q, 11</td>
<td>INPUT RETURN CHARACTER</td>
</tr>
<tr>
<td>1315</td>
<td>TIE FLAGS, 10000 (\text{IIS TEMP, MODE FLAG SET})</td>
<td></td>
</tr>
<tr>
<td>1316</td>
<td>IYES, JUMP</td>
<td></td>
</tr>
<tr>
<td>1317</td>
<td>TIE FLAGS, 1100 (\text{CLEAR PHONIC ON FLAGS})</td>
<td></td>
</tr>
<tr>
<td>1318</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1319</td>
<td>TIE FLAGS, 1100 (\text{YES, PHONIC ON})</td>
<td></td>
</tr>
<tr>
<td>1320</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
<tr>
<td>1321</td>
<td>TIE FLAGS, 1100 (\text{CLEAR TEM, PHONIC ON FLAG})</td>
<td></td>
</tr>
<tr>
<td>1322</td>
<td>CAIN B, &quot;A&quot;</td>
<td>IS CHARACTER IN AC-B (&quot;A&quot;)</td>
</tr>
<tr>
<td>1323</td>
<td>TIE FLAGS, 1100 (\text{SET TEMP, PHONIC ON})</td>
<td></td>
</tr>
<tr>
<td>1324</td>
<td>JRST EVAL2</td>
<td>GO TO EVAL2 (SECTION I)</td>
</tr>
</tbody>
</table>
SECTION U. EXTRA ON ORDER ON COMPARISON SUBROUTINE,

FRSUB1: MOVEM G,PC2
ISAVE RETURN ADDRESS

TLZ FLGS,6,
ICLEAR END OF ANSWER FLAGS

FRSUB1: MOVEM G,PC2
ICLEAR NO MATCH AND EXTRA WORD FLAGS

JSP Q,STDPNT
IGET STUDENT ANSWER POINTER (SECTION X)

JSP Q,STPNT
ADVANCE TO BEGINNING OF FIRST WORD (SECTION X)

MOVEM B,SAVPNT
ISAVE THE ANSWER

JSP Q,LESPT
IGET THE LESSON ANSWER POINTER (SECTION X)

JSP Q,STPNTS
ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)

JST FR1
I GO TO FR1

MOVEM G,PC2
ISAVE RETURN ADDRESS

TLZ FLGS,6,
ICLEAR END OF ANSWER FLAGS

FRSUB1: MOVEM G,PC2
ICLEAR NO MATCH AND EXTRA WORD FLAGS

SETMH COUNT+I
ISET NO. OF WORDS MATCHED TO ZERO

JSP Q,LESPT
IGET LESSON ANSWER POINTER (SECTION X)

JSP Q,STPNT
ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)

MOVEM B,SAVPNT
ISAVE THE ANSWER

JSP Q,STPNTS
ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)

TLNN FLGS,4
IEND OF LESSON ANSWER?

JST +I
I ND, SKIP

TLNE FLGS,2
IEND OF STUDENT ANSWER?

JST GRANS
IYES, GO TO GRANS (SECTION G)

MOVEM G,PC2
IISAVE RETURN ADDRESS

JST 12
IRETURN

TLNE FLGS,2
IEND OF STUDENT ANSWER?

JST FR1
IYES, GO TO FR1 (SECTION I)

EXCH B,SAVPNT
IXCHANGE POINTERS

MOVEM B,SAVPNT
IXCHANGE LESSON WORD HOLDER

JSP Q,LESPT
ICLEAR THE LESSON WORD HOLDER (SECTION X)

JSP Q,STPNT
ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)

TLNN FLGS,4
IEND OF LESSON ANSWER?

JST CHKcnt
IYES, GO TO CHKcnt

MOVEM G,PC2
ISAVE CURRENT STUDENT ANSWER POINTER

MOVEM B,SAVPNT+I
IXCHANGE POINTERS

MOVEM Q,PC2
IISAVE RETURN ADDRESS

JST FR1
IEND OF MATCH COUNT ZERO?

I YES, SKIP

MOVEM B,PC2
ISAVE RETURN ADDRESS

SKIPE COUNT
IIS REQUIRED MATCH COUNT ZERO?

JST +1
I IND, SKIP
SECTION V. EXTRA ON, ORDER OFF COMPARISON SUBROUTINE:

MOVE M.0,PC2 ISAVE RETURN ADDRESS
TLZ_FLAGS,8 ICLRC GREAT RETURN ADDRESS
TIN_FLAGS,100000 ICLRC GREAT RETURN ADDRESS
SETN, COUNT+1 SET NO. OF WORDS REQUIRED FOR MATCH TO ZERO
JSP M,LESPNT GET LESSON ANSWER POINTER (SECTION X)
JSP M,STPNL ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)
MOVE M.SAVPNT ISAVE POINTER
JSP M,STPNL GET STUDENT ANSWER POINTER (SECTION X)
JSP M,STPNL ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)
TLZ_FLAGS,4 IEND OF LESSON ANSWER?
JST NOTEND INO. GO TO NOTEND
TLZ_FLAGS,2 IEND OF STUDENT ANSWER?
JST GTRANS YES, GOTO GTRANS (SECTION G)
JST 1,44 YES, JUMP
TLZ_FLAGS,420000 E X E R R O R M O D E?
JST GTRANS YES, GOTO GTRANS (SECTION G)
JST 1,42 E X E R R O R M O D E?
JST NOTEND IEND OF STUDENT ANSWER?
JST EVAL YES, GOTO EVAL (SECTION I)
FRES1 EXCH M.SAVPNT EXCHANGE POINTERS
MOVE M.POINTS GET POINTER TO LESSON WORD HOLDER
JSP M,LESPNT ICLRC THE LESSON WORD HOLDER (SECTION X)
JSP M,STPNL ICLRC THE LESSON WORD HOLDER (SECTION X)
TLZ_FLAGS,4 ISET Flag OF END OF LESSON ANSWER
JSP M,STPNL ADVANCE TO BEGINNING OF NEXT WORD (SECTION X)
<table>
<thead>
<tr>
<th>Address</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>001701</td>
<td>\texttt{FREQ1 EXCH B,SAVPNT}</td>
</tr>
<tr>
<td>001703</td>
<td>\texttt{MOVE C,POINT8}</td>
</tr>
<tr>
<td>001705</td>
<td>\texttt{GET POINTER TO STUDENT WORD HOLDER}</td>
</tr>
<tr>
<td>001707</td>
<td>\texttt{JSP Q,2EROS}</td>
</tr>
<tr>
<td>001709</td>
<td>\texttt{CLEAR WORD HOLDER (SECTION X)}</td>
</tr>
<tr>
<td>001711</td>
<td>\texttt{JSP Q,LDWDO}</td>
</tr>
<tr>
<td>001713</td>
<td>\texttt{LOAD WORD INTO WORD HOLDER (SECTION X)}</td>
</tr>
<tr>
<td>001715</td>
<td>\texttt{TLO FLAGS,0}</td>
</tr>
<tr>
<td>001717</td>
<td>\texttt{SET FLAG IF END OF STUDENT ANSWER}</td>
</tr>
<tr>
<td>001719</td>
<td>\texttt{JSP Q,STPNTS}</td>
</tr>
<tr>
<td>00171B</td>
<td>\texttt{ADVANCE BEGINNING OF NEXT WORD (SECTION X)}</td>
</tr>
<tr>
<td>00171D</td>
<td>\texttt{JSP Q,CMMPAR}</td>
</tr>
<tr>
<td>00171F</td>
<td>\texttt{COMPARE LESSON WORD TO STUDENT WORD (SECTION X)}</td>
</tr>
<tr>
<td>001721</td>
<td>\texttt{JST NHTCH2}</td>
</tr>
<tr>
<td>001723</td>
<td>\texttt{IF NO MATCH, GO TO NHTCH2}</td>
</tr>
<tr>
<td>001725</td>
<td>\texttt{AOS COUNT+1}</td>
</tr>
<tr>
<td>001727</td>
<td>\texttt{ELSE, ADD ONE TO NO. OF WORDS MATCHED}</td>
</tr>
<tr>
<td>001729</td>
<td>\texttt{TNLN FLAGS,4}</td>
</tr>
<tr>
<td>00172B</td>
<td>\texttt{END OF LESSON ANSWER?}</td>
</tr>
<tr>
<td>00172D</td>
<td>\texttt{JSP Q,STPNT}</td>
</tr>
<tr>
<td>00172F</td>
<td>\texttt{GET INITIAL STUDENT ANSWER POINTER (SECTION X)}</td>
</tr>
<tr>
<td>001731</td>
<td>\texttt{JSP Q,STPNTS}</td>
</tr>
<tr>
<td>001733</td>
<td>\texttt{ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)}</td>
</tr>
<tr>
<td>001735</td>
<td>\texttt{NMCH2}</td>
</tr>
<tr>
<td>001737</td>
<td>\texttt{TNLN FLAGS,4}</td>
</tr>
<tr>
<td>001739</td>
<td>\texttt{END OF LESSON ANSWER?}</td>
</tr>
<tr>
<td>00173B</td>
<td>\texttt{JST NTSET2}</td>
</tr>
<tr>
<td>00173D</td>
<td>\texttt{IND. GO TO NTSET2}</td>
</tr>
<tr>
<td>00173F</td>
<td>\texttt{TNLN FLAGS,2}</td>
</tr>
<tr>
<td>001741</td>
<td>\texttt{END OF STUDENT ANSWER?}</td>
</tr>
<tr>
<td>001743</td>
<td>\texttt{JST FRE2}</td>
</tr>
<tr>
<td>001745</td>
<td>\texttt{IND. GO TO FRE2}</td>
</tr>
<tr>
<td>001747</td>
<td>\texttt{TRO FLAGS,10000}</td>
</tr>
<tr>
<td>001749</td>
<td>\texttt{ELSE, SET NO MATCH FLAG}</td>
</tr>
<tr>
<td>00174B</td>
<td>\texttt{JST CHKN2}</td>
</tr>
<tr>
<td>00174D</td>
<td>\texttt{GO TO CHKN2}</td>
</tr>
<tr>
<td>00174F</td>
<td>\texttt{CHKN2}</td>
</tr>
<tr>
<td>001751</td>
<td>\texttt{MOVE Q,FPC2}</td>
</tr>
<tr>
<td>001753</td>
<td>\texttt{GET RETURN ADDRESS}</td>
</tr>
<tr>
<td>001755</td>
<td>\texttt{SKIP COUNT}</td>
</tr>
<tr>
<td>001757</td>
<td>\texttt{JUST MUST ALL WORDS MATCH?}</td>
</tr>
<tr>
<td>001759</td>
<td>\texttt{JST FRE2}</td>
</tr>
<tr>
<td>00175B</td>
<td>\texttt{IND. GO TO FRE2}</td>
</tr>
<tr>
<td>00175D</td>
<td>\texttt{TRO FLAGS,10000}</td>
</tr>
<tr>
<td>00175F</td>
<td>\texttt{ELSE, SET NO MATCH FLAG}</td>
</tr>
<tr>
<td>001761</td>
<td>\texttt{JSP Q,STPNT}</td>
</tr>
<tr>
<td>001763</td>
<td>\texttt{GET INITIAL STUDENT ANSWER POINTER (SECTION X)}</td>
</tr>
<tr>
<td>001765</td>
<td>\texttt{JSP Q,STPNTS}</td>
</tr>
<tr>
<td>001767</td>
<td>\texttt{ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)}</td>
</tr>
<tr>
<td>001769</td>
<td>\texttt{JST FRE1}</td>
</tr>
<tr>
<td>00176B</td>
<td>\texttt{GO TO FRE1}</td>
</tr>
<tr>
<td>00176D</td>
<td>\texttt{CHKN2}</td>
</tr>
<tr>
<td>00176F</td>
<td>\texttt{MOVE Q,FPC2}</td>
</tr>
<tr>
<td>001771</td>
<td>\texttt{GET RETURN ADDRESS}</td>
</tr>
<tr>
<td>001773</td>
<td>\texttt{SKIP COUNT}</td>
</tr>
<tr>
<td>001775</td>
<td>\texttt{JUST MUST ALL WORDS MATCH?}</td>
</tr>
<tr>
<td>001777</td>
<td>\texttt{JST FRE1}</td>
</tr>
<tr>
<td>001779</td>
<td>\texttt{IND. GO TO FRE1}</td>
</tr>
<tr>
<td>00177B</td>
<td>\texttt{TRO FLAGS,10000}</td>
</tr>
<tr>
<td>00177D</td>
<td>\texttt{ELSE, SET NO MATCH FLAG}</td>
</tr>
<tr>
<td>00177F</td>
<td>\texttt{JSP Q,STPNT}</td>
</tr>
<tr>
<td>001781</td>
<td>\texttt{GET INITIAL STUDENT ANSWER POINTER (SECTION X)}</td>
</tr>
<tr>
<td>001783</td>
<td>\texttt{JSP Q,STPNTS}</td>
</tr>
<tr>
<td>001785</td>
<td>\texttt{ADVANCE TO BEGINNING OF 1ST WORD (SECTION X)}</td>
</tr>
<tr>
<td>001787</td>
<td>\texttt{JST FRE1}</td>
</tr>
<tr>
<td>001789</td>
<td>\texttt{GO TO FRE1}</td>
</tr>
</tbody>
</table>

**SECTION X:** PHONETICALLY PROCESS THE STUDENT’S ANSWER.
SECTION X. THE SUBROUTINES BELOW ARE CALLED BY THE ANSWER EVALUATION SUBROUTINES.
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>MOV G.POINTS</td>
<td>PUT LESSON ANSWER POINTER IN AC-G</td>
</tr>
<tr>
<td>11</td>
<td>MOVE INDEX,IDXSAV</td>
<td>SAVE DATA INDEX</td>
</tr>
<tr>
<td>12</td>
<td>JSP 2,POPLST</td>
<td>EVALUATE LESSON ANSWER</td>
</tr>
<tr>
<td>13</td>
<td>MOVE INDEX,IDXSAV</td>
<td>RESTORE DATA INDEX</td>
</tr>
<tr>
<td>14</td>
<td>PCD 2,AC</td>
<td>SET AC-D TO ZERO TO INDICATE NO UNITS</td>
</tr>
<tr>
<td>15</td>
<td>CAIN A, &quot;H&quot;</td>
<td>DID A CONMA FOLLOW THE ANSWER?</td>
</tr>
<tr>
<td>16</td>
<td>SETO D,</td>
<td>SET AC-D TO ONES TO INDICATE UNITS</td>
</tr>
<tr>
<td>17</td>
<td>MOVE A,POPLST</td>
<td>PUT VALUE OF ANSWER IN AC-A</td>
</tr>
<tr>
<td>18</td>
<td>SKIP IF NOT TOLERANCE IS SPECIFIED</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>ITOTLCHK</td>
<td>IF CHECK IF ANSWER IS WITHIN TOLERANCE</td>
</tr>
<tr>
<td>20</td>
<td>CAEN ASANS</td>
<td>IS LESSON ANSWER EQUAL TO STUDENT ANSWER?</td>
</tr>
<tr>
<td>21</td>
<td>JRTST EVAL</td>
<td>NO, GO TO EVAL (SECTION I)</td>
</tr>
<tr>
<td>22</td>
<td>JUMP D, STANS</td>
<td>IF NO UNITS GO TO STANS (SECTION G)</td>
</tr>
<tr>
<td>23</td>
<td>JRTST COMPL</td>
<td>ELSE, GO CHECK UNITS</td>
</tr>
<tr>
<td>24</td>
<td>TOLKCH</td>
<td>SUBTRACT STUDENT ANSWER FROM LESSON ANSWER</td>
</tr>
<tr>
<td>25</td>
<td>MOVHMS A</td>
<td>GET ABSOLUTE VALUE OF THE DIFFERENCE</td>
</tr>
<tr>
<td>26</td>
<td>MOVHMS TOL</td>
<td>TAKE ABSOLUTE VALUE OF TOLERANCE</td>
</tr>
<tr>
<td>27</td>
<td>CAEN A,O</td>
<td>IS ANSWER WITHIN TOLERANCE?</td>
</tr>
<tr>
<td>28</td>
<td>JRTST EVAL</td>
<td>NO, GO TO EVAL (SECTION I)</td>
</tr>
<tr>
<td>29</td>
<td>JRTST NUMER2</td>
<td>YES, GO SEE IF UNITS ARE PRESENT</td>
</tr>
</tbody>
</table>

SECTION 2: FIND THE ACTIONS FOR THE MATCHED ANSWER.
ISECTION EE, INTERPRET = FRAME

2245 003063 622 00 00 000110 RFRAFET BEEFFLAGS,100 HAS REVIEW REQUESTED?
2246 003064 254 00 00 003007 JSRT *+3 YES, JUMP
2247 003065 682 00 00 000250 TWEFFLAGS,200 IS REVIEW IN PROGRESS?
2248 003066 254 00 00 003124 JSRT CONTN2 YES, GO TO CONTN2 (SECTION FF)
2249 003067 220 00 00 000466 MOVE D,REVIDX IS REVIEW TABLE INDEX
2250 003068 322 00 00 003212 JUMPE D,STRNUM IF 1ST REVIEW FRAME, GO TO STRNUM
2251 003069 200 00 00 002654 MOVE B,FRTNUM GET CURRENT FRAME NO.
2252 00306a 301 00 00 001759 CALL B, +1000 IS REVIEW IN PROGRESS?
2253 00306b 254 00 00 003052 JSRT RF3 YES, GO TO RF2
2254 00306c 361 00 00 003597 SICL D,STRNUM STORE REVIEW Frame POINTER IF NOT YET STORED
2255 00306d 135 00 00 000012 MOVE A, PONT11 GET CURRENT FRAME NO.
2256 00306e 316 00 00 000022 JSRT RF2 YES, GO TO RF2
2257 00306f 254 00 00 003052 JSRT RF3 YES, GO TO RF2
2258 003070 254 00 00 003014 JSRT RF1 ELSE, GO CHECK NEXT REVIEW TABLE ENTRY
2259 003071 660 00 00 003400 JSRT RF11 ELSE, GO CHECK NEXT REVIEW TABLE ENTRY
2260 003072 200 00 00 003307 MOVE D,REVIDX GET REVIEW TABLE INDEX
2261 003073 200 00 00 003011 MOVE A,FRTNUM PUT FRAME NO. IN AC-A
2262 003074 137 00 00 000012 JSRT RF2 TRANSFER IT TO REVIEW TABLE
2263 003075 301 00 00 000011 JSP S, +11 INPUT CHARACTER
2264 003076 301 00 00 000027 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2265 003077 254 00 00 003025 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2266 003078 265 17 00 000016 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2267 003079 137 00 00 000043 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2268 00307a 392 00 00 003022 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2269 00307b 392 00 00 003022 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2270 00307c 051 00 00 003721 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2271 00307d 366 00 00 000012 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2272 00307e 254 00 00 003041 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2273 00307f 271 00 00 000065 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2274 003080 391 00 00 000021 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2275 003081 254 00 00 003041 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2276 003082 051 00 00 003721 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2277 003083 051 00 00 000016 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2278 003084 265 17 00 000011 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
2279 003085 391 00 00 003721 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
227a 003086 392 00 00 003022 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
227b 003087 051 00 00 003721 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
227c 003088 051 00 00 000016 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
227d 003089 265 17 00 000011 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
227e 00308a 391 00 00 003721 JSRT RF7 BEGINNING OF REVIEW FRAME POINTER?
<table>
<thead>
<tr>
<th>Line</th>
<th>Hex</th>
<th>Decimal</th>
<th>ASCII</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2280</td>
<td>003845'</td>
<td>00 00 00 00 33 00</td>
<td>JRST EOL</td>
<td>YES, GO TO EOL (SECTION HH)</td>
</tr>
<tr>
<td>2281</td>
<td>003846'</td>
<td>00 00 00 00 33 00</td>
<td>CAIN A.36</td>
<td>BEGINNING OF NEXT FRAME</td>
</tr>
<tr>
<td>2282</td>
<td>003847'</td>
<td>00 00 00 00 33 00</td>
<td>JRST FSTLIN</td>
<td>YES, GO TO FSTLIN (SECTION B)</td>
</tr>
<tr>
<td>2283</td>
<td>003848'</td>
<td>00 00 00 00 33 00</td>
<td>TYCALL 1,A</td>
<td>ELSE, TYPE THE CHARACTER</td>
</tr>
<tr>
<td>2284</td>
<td>003849'</td>
<td>00 00 00 00 33 00</td>
<td>JRST RF3</td>
<td>GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>2285</td>
<td>00384A'</td>
<td>00 00 00 00 33 00</td>
<td>JSF OP:11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>2286</td>
<td>00384B'</td>
<td>00 00 00 00 33 00</td>
<td>CAIN A.4</td>
<td>END OF LESSON?</td>
</tr>
<tr>
<td>2287</td>
<td>00384C'</td>
<td>00 00 00 00 33 00</td>
<td>JRST RF3</td>
<td>YES, GO TO EOL (SECTION HH)</td>
</tr>
<tr>
<td>2288</td>
<td>00384D'</td>
<td>00 00 00 00 33 00</td>
<td>CAIN A.36</td>
<td>BEGINNING OF NEXT FRAME</td>
</tr>
<tr>
<td>2289</td>
<td>00384E'</td>
<td>00 00 00 00 33 00</td>
<td>JRST FSTLIN</td>
<td>YES, GO TO FSTLIN (SECTION B)</td>
</tr>
<tr>
<td>2290</td>
<td>00384F'</td>
<td>00 00 00 00 33 00</td>
<td>JRST RF2</td>
<td>ELSE, GO INPUT NEXT CHARACTER</td>
</tr>
</tbody>
</table>

### Section FF: Execute Student Review Commands

```plaintext
2292

2295 | 003850' | 00 00 00 00 33 00 | REVIEW SET D. |
| 2296 | 003851' | 00 00 00 00 33 00 | JRST 1.2 | INPUT DIGIT |
| 2297 | 003852' | 00 00 00 00 33 00 | RBI | |
| 2298 | 003853' | 00 00 00 00 33 00 | CAIN A.10 | END OF NUMBER? |
| 2299 | 003854' | 00 00 00 00 33 00 | JRST RNIN | YES, GO TO RNIN |
| 2300 | 003855' | 00 00 00 00 33 00 | CAIL A.10 | |
| 2301 | 003856' | 00 00 00 00 33 00 | JRST RNIN | YES, GO TO RNIN |
| 2302 | 003857' | 00 00 00 00 33 00 | IMULT D:12 | MULTIPLY NO. IN AC-0 BY TEN |
| 2303 | 003858' | 00 00 00 00 33 00 | Addr 1,12 | ADD VALUE OF DIGIT TO NO. IN AC-0 |
| 2304 | 003859' | 00 00 00 00 33 00 | JRST RB | GO INPUT NEXT DIGIT |
| 2305 | 00385A' | 00 00 00 00 33 00 | RNIN | TCCALL 4,A | INPUT LINE FEED |
| 2306 | 00385B' | 00 00 00 00 33 00 | JUMP D:RNERR | IF ERROR, GO TYPE ERROR MESSAGE |
| 2307 | 00385C' | 00 00 00 00 33 00 | CAIL A.12 | IS NO. GREATER THAN 10? |
| 2308 | 00385D' | 00 00 00 00 33 00 | JRST RNERR | YES, GO TYPE ERROR MESSAGE |
| 2309 | 00385E' | 00 00 00 00 33 00 | SOR D | SUBTRACT TO GET REVIEW TABLE INDEX |
| 2310 | 00385F' | 00 00 00 00 33 00 | Addr 0,REVINDX | HAS THAT TOPIC BEEN ENCOUNTERED? |
| 2311 | 003860' | 00 00 00 00 33 00 | JRST TO15 | NO. GO TYPE ERROR MESSAGE |
| 2312 | 003861' | 00 00 00 00 33 00 | Addr 0 | SOR INDEX | BACKUP DATA INDEX |
| 2313 | 003862' | 00 00 00 00 33 00 | Addr 0,REVINDX | GET CURRENT FRAME NO. |
| 2314 | 003863' | 00 00 00 00 33 00 | Addr 0,REVINDX | IS THIS A REVIEW FRAME? |
| 2315 | 003864' | 00 00 00 00 33 00 | Addr 0,REVINDX | ADD TO FRAME NO. |
| 2316 | 003865' | 00 00 00 00 33 00 | Addr 0,REVINDX | STORE FRAME NO. WITH STUD. PERF. DATA |
| 2317 | 003866' | 00 00 00 00 33 00 | Addr 0,REVINDX | RESE TO INDEX |
| 2318 | 003867' | 00 00 00 00 33 00 | Addr 0,REVINDX | SET REVIEW IN PROGRESS AND REVIEW SLIDE FLAGS |
| 2319 | 003868' | 00 00 00 00 33 00 | Addr 0,REVINDX | PUT POINTER TO REVIEW POINT IN AC-0 |
| 2320 | 003869' | 00 00 00 00 33 00 | Addr 0,REVINDX | SET REVIEW REQUESTED FLAG |
| 2321 | 00386A' | 00 00 00 00 33 00 | Addr 0,REVINDX | GO TO THE BRANCHING ROUTINE (ACTION.HAC) |
| 2322 | 00386B' | 00 00 00 00 33 00 | TCCALL 3,CASIZE | YOU HAVE NOT YET PASSED THAT REVIEW POINT, ANSWER THE QUESTION OR |
| 2323 | 00386C' | 00 00 00 00 33 00 | Addr 0,REVINDX | CHOOSE ANOTHER REVIEW POINT. |
| 2324 | 00386D' | 00 00 00 00 33 00 | Addr 0,REVINDX | MOVE TO ANOTHER REVIEW INDEX |
| 2325 | 00386E' | 00 00 00 00 33 00 | Addr 0,REVINDX | JUMP TO ANOTHER REVIEW INDEX |
| 2326 | 00386F' | 00 00 00 00 33 00 | Addr 0,REVINDX | JRST WAIT | ELSE, CONTINUE TO WAIT FOR ANSWER (SECTION G) |
```
JST EDIT
igo get the lesson editor

TSTL:
input next digit

JST TST2
if end of frame no., go to TST2

IMUL D.12
multiply no. in AC-D by 10

ADDI D,-69(A)
add value of digit to no. in AC-D

JST TST1
input next digit

CALL 11.9
clear tty input buffer

TST21
if has desired frame been passed?

JST TST3
ind. go to TST3

USEI 1.1
else return to beginning of lesson

INPUT 1
input 1st disk block

MOVE 14,1BF=1
input byte pointer in AC-14

MOVE 13,1BF=2
input byte count in AC-15

MOVE A,1

MOVE A,DSKBLK
set disk block to 1

JSP Q,11
input character

CAIE A,1
text statement?

JSP Q,SKPXT
else skip the text (section A)

JST TST3
ind. go to TST3

CAIE A,4
end of lesson?

JST TST5
yes, go to TST5

CAIE A,36
beginning of frame?

JST TST3
ind. go to input next character

MOVE A,DSKBLK
save disk block no.

MOVE 14,1BF=2
isave buffer byte pointer

MOVE 15,1BF=2
isave byte count

TRNE FLAGS.208
is review in progress?

JST TST3
ind. skip

MOVE A,SAVPOS
else, save disk block no.

MOVE 14,SAVPOS+1
isave buffer byte pointer

MOVE 15,SAVPOS+2
isave byte count

JSP Q.110
input frame no.

JST TST5
ind. go to next character

GAME B.D
is this the desired frame?

JST TST3
ind. go to input next character

ACS INDEX
increment data index

JSP Q,EXPCOR
expand core if necessary

MOVE B.0
input frame no., in AC-D

MOVE A,FRMTYP
input frame type in AC-A

JST FSTL2
ind. go to FSTL2 (section B)

TST5:
tcall 3,(ASC12) / frame not found, try again, */

MOVE B,10000

MOVE B,FRNMUL
move larger than allowed frame no. to FRNMUL

JST TSTM
ind. accept another frame no.

EDIT1
move A,ARGMNT
input address of getseg arguments in AC-A

MOVE B,6
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2545</td>
<td>023414</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2546</td>
<td>023415</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2547</td>
<td>023416</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2548</td>
<td>023417</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2549</td>
<td>023418</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2550</td>
<td>023419</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2551</td>
<td>023420</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2552</td>
<td>023421</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2553</td>
<td>023422</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2554</td>
<td>023423</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2555</td>
<td>023424</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2556</td>
<td>023425</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2557</td>
<td>023426</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2558</td>
<td>023427</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2559</td>
<td>023428</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2560</td>
<td>023429</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2561</td>
<td>023430</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2562</td>
<td>023431</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2563</td>
<td>023432</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2564</td>
<td>023433</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2565</td>
<td>023434</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2566</td>
<td>023435</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2567</td>
<td>023436</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2568</td>
<td>023437</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2569</td>
<td>023438</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2570</td>
<td>023439</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2571</td>
<td>023440</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2572</td>
<td>023441</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2573</td>
<td>023442</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2574</td>
<td>023443</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2575</td>
<td>023444</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2576</td>
<td>023445</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2577</td>
<td>023446</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2578</td>
<td>023447</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2579</td>
<td>023448</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2580</td>
<td>023449</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2581</td>
<td>023450</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2582</td>
<td>023451</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2583</td>
<td>023452</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2584</td>
<td>023453</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2585</td>
<td>023454</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2586</td>
<td>023455</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2587</td>
<td>023456</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2588</td>
<td>023457</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2589</td>
<td>023458</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2590</td>
<td>023459</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2591</td>
<td>023460</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2592</td>
<td>023461</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2593</td>
<td>023462</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2594</td>
<td>023463</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2595</td>
<td>023464</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2596</td>
<td>023465</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
<tr>
<td>2597</td>
<td>023466</td>
<td>EXECUTE MACRO 47(204)-7 200106 2-AUG-74 PAGE 1-48</td>
</tr>
</tbody>
</table>

**Notes:**
- This is a listing of assembly language code, likely for a computer program.
- The comments indicate the purpose of each line of code.
- The code appears to be written in a format typical of older computer languages or systems.
INTERP MACRO 47(204)-9 00106 2-AUG-74 PAGE 1-50

2651 0035671 205 01 0 0 0 447000
2652 0235721 202 01 0 0 0 0 00002*
2653 0235711 492 00 0 0 0 00003*
2654 0235721 055 01 0 0 0 003557*
2655 0235731 255 00 0 0 0 000000
2656
2657
2658 0235741 607 00 0 0 0 000001
2659 0235751 254 00 0 0 0 003313*
2660 0235761 254 00 0 0 0 000000*
2661
2662

NO ERRORS DETECTED

PROGRAM BREAK IS 004031

4K CORE USED
<table>
<thead>
<tr>
<th>QCMD</th>
<th>2458#</th>
<th>2539</th>
<th>571</th>
<th>404</th>
<th>832</th>
<th>619</th>
<th>740</th>
<th>719</th>
<th>316</th>
<th>242</th>
<th>2089</th>
<th>571</th>
<th>571</th>
<th>571</th>
<th>571</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCMQ</td>
<td>425#</td>
<td>2258</td>
<td>2458</td>
<td>2158</td>
<td>2175</td>
<td>2178</td>
<td>2638</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUTQ</td>
<td>1291#</td>
<td>2158</td>
<td>2158</td>
<td>2175</td>
<td>2178</td>
<td>2638</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIQ</td>
<td>1654#</td>
<td>1639#</td>
<td>1642</td>
<td>1665</td>
<td>1706</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIQ</td>
<td>1653#</td>
<td>1715</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2Q</td>
<td>175#</td>
<td>1707</td>
<td>1728</td>
<td>1732</td>
<td>1740</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIQ</td>
<td>1751#</td>
<td>1722</td>
<td>1728</td>
<td>1732</td>
<td>1740</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAG</td>
<td>416</td>
<td>747</td>
<td>752</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAGQ</td>
<td>645</td>
<td>747</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RQ</td>
<td>1291#</td>
<td>1291#</td>
<td>1291#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ1</td>
<td>1359#</td>
<td>1359#</td>
<td>1359</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ1</td>
<td>448</td>
<td>955</td>
<td>960</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ2</td>
<td>449</td>
<td>953</td>
<td>960</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ3</td>
<td>450</td>
<td>1105</td>
<td>1728</td>
<td>1732</td>
<td>2178</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ4</td>
<td>451</td>
<td>597</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ5</td>
<td>452</td>
<td>597</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ6</td>
<td>453</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ7</td>
<td>454</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ8</td>
<td>455</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ9</td>
<td>456</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPIQ10</td>
<td>457</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ13</td>
<td>465</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPNQ1</td>
<td>427</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPNQ2</td>
<td>428</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ1</td>
<td>457</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ2</td>
<td>458</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ3</td>
<td>459</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ4</td>
<td>460</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ5</td>
<td>461</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ6</td>
<td>462</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ7</td>
<td>463</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ8</td>
<td>464</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ9</td>
<td>465</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ10</td>
<td>466</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ11</td>
<td>467</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ12</td>
<td>468</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PONQ13</td>
<td>469</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QCMD</td>
<td>428</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFRAH1</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFRAME</td>
<td>639</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNUM</td>
<td>425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RMD</td>
<td>2297#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RCMQ</td>
<td>428#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECDAT</td>
<td>2503</td>
<td>2514#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECDAT</td>
<td>663#</td>
<td>662#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECNT1</td>
<td>2621#</td>
<td>2505#</td>
<td>2513</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECNT2</td>
<td>2622#</td>
<td>2648</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECNH2</td>
<td>2623#</td>
<td>2649#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETFND</td>
<td>1782</td>
<td>1798#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETHBD</td>
<td>429#</td>
<td>2088</td>
<td>2085</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REV1</td>
<td>941#</td>
<td>1928#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REV2</td>
<td>423#</td>
<td>572</td>
<td>2249</td>
<td>2260</td>
<td>2266</td>
<td>2311</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REV4</td>
<td>1256#</td>
<td>1259#</td>
<td>2295#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REVNUM</td>
<td>422#</td>
<td>458</td>
<td>459</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REVRS</td>
<td>791#</td>
<td>634</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REVSET</td>
<td>661#</td>
<td>688#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF1</td>
<td>2251#</td>
<td>2259</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF2</td>
<td>2253</td>
<td>2257</td>
<td>2285#</td>
<td>2290</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF3</td>
<td>2278#</td>
<td>2284</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFRAFTE</td>
<td>617#</td>
<td>2245#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFCNT</td>
<td>619#</td>
<td>468</td>
<td>984</td>
<td>1197</td>
<td>1198</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNAT1</td>
<td>2392</td>
<td>2337</td>
<td>2509</td>
<td>2338#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNAT2</td>
<td>2299</td>
<td>2306#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNIP</td>
<td>2354</td>
<td>2366#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNMP</td>
<td>429#</td>
<td>514</td>
<td>2193</td>
<td>2550</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPTCNT</td>
<td>423#</td>
<td>637</td>
<td>1104</td>
<td>1263</td>
<td>1951</td>
<td>1980</td>
<td>2039</td>
<td>2060</td>
<td>2061</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSNM</td>
<td>422#</td>
<td>533#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTN</td>
<td>476#</td>
<td>495</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RURNTM</td>
<td>429#</td>
<td>485</td>
<td>2569</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SANS</td>
<td>428#</td>
<td>457</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAVPNT</td>
<td>425#</td>
<td>1434</td>
<td>1437</td>
<td>1443</td>
<td>1493</td>
<td>1503</td>
<td>1514</td>
<td>1520</td>
<td>1521</td>
<td>1555</td>
<td>1568</td>
<td>1585</td>
<td>1591</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAVPOS</td>
<td>429#</td>
<td>582</td>
<td>563</td>
<td>584</td>
<td>590</td>
<td>596</td>
<td>2557</td>
<td>2535</td>
<td>2361</td>
<td>2363</td>
<td>2420</td>
<td>2421</td>
<td>2422</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMO</td>
<td>429#</td>
<td>2091</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETBIT</td>
<td>675#</td>
<td>677</td>
<td>679</td>
<td>681</td>
<td>693#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETBMT</td>
<td>704#</td>
<td>714</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKEST</td>
<td>794#</td>
<td>997</td>
<td>1000</td>
<td>1007</td>
<td>1211#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKPTXT</td>
<td>416</td>
<td>1185</td>
<td>2125</td>
<td>2104</td>
<td>1211#</td>
<td>2120</td>
<td>2124</td>
<td>2408</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLBDRTG</td>
<td>823#</td>
<td>832</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLDFD</td>
<td>882#</td>
<td>834#</td>
<td>843</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLTG</td>
<td>426#</td>
<td>518</td>
<td>819</td>
<td>821</td>
<td>830</td>
<td>831</td>
<td>841</td>
<td>842</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLDNCH</td>
<td>846#</td>
<td>858#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLDO</td>
<td>814#</td>
<td>854#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLONRT</td>
<td>416</td>
<td>793#</td>
<td>796#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLRMT</td>
<td>852#</td>
<td>862#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLTCH</td>
<td>847</td>
<td>793#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLTCH</td>
<td>845#</td>
<td>871</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNA</td>
<td>421#</td>
<td>1247</td>
<td>1920</td>
<td>1924</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSS</td>
<td>1694</td>
<td>1718#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>1641</td>
<td>1659#</td>
<td>1856</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STDAT</td>
<td>1488#</td>
<td>487</td>
<td>2541</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STDIM</td>
<td>1343</td>
<td>1355#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STDDT</td>
<td>509#</td>
<td>1353</td>
<td>1361#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STDPM2</td>
<td>1768</td>
<td>1770</td>
<td>1774</td>
<td>1776#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STDPMT</td>
<td>1432</td>
<td>1491</td>
<td>1504</td>
<td>1569</td>
<td>1683</td>
<td>1615</td>
<td>1763#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CALC MACRO EXECUTES C-FRAME STATEMENTS AND EVALUATES POLISH EXPRESSIONS.

CONTENTS OF CALC.MAC

SECTION A. CONTAINS ENTRY POINTS TO CALC.MAC.
SECTION B. PROCESSES ITERATION FIELDS.
SECTION C. LOADS CALC STATEMENTS INTO STATEMENT BUFFER.
SECTION D. INCREMENTS ITERATION INDICES.
SECTION E. DETERMINES STATEMENT TYPE.
SECTION F. EXECUTES INPUT STATEMENTS.
SECTION G. EXECUTES ATYPE STATEMENTS.
SECTION H. EXECUTES IF STATEMENTS.
SECTION I. EXECUTES BI STATEMENTS.
SECTION J. EXECUTES G1 STATEMENTS.
SECTION K. EXECUTES GOTO STATEMENTS.
SECTION L. EXECUTES FILL STATEMENTS.
SECTION M. EXECUTES TYPE STATEMENTS.
SECTION N. EXECUTES ASSIGNMENT STATEMENTS.
SECTION O. EVALUATES POLISH STRINGS.
SECTION P. EXECUTES CALL STATEMENTS.
SECTION Q. EXECUTES RETURN STATEMENTS.
ENTRY POINTS TO CALC.MAC
CALC = FROM INTERP,MAC (SECTION H), LOCATED IN SECTION A

CALCM = FROM LEARN,MAC (SECTION H), LOCATED IN SECTION A

CFRAME = FROM INTERP,MAC (SECTION B), LOCATED IN SECTION A

CFRAME*1 = FROM INTERP,MAC (SECTIONS E & K) AND ACTION,MAC
(SECTION A, C, D, E, F, H & I), LOCATED IN SECTION A

POLISH = FROM INTERP,MAC (SECTION Y) AND ACTION,MAC (SECTION E),
LOCATED IN SECTION G

EXTERNAL SUBROUTINES CALLED BY CALC,MAC

BCHD2 = EXECUTES BRANCHING (B1), LOCATED IN ACTION,MAC (SECTION K)

CPROS = PROCESSES CALC MODE STATEMENTS, LOCATED IN CPROS,MAC
(SECTION A)

FABS = COMPUTES ABSOLUTE VALUE OF NUMBER IN AC=A, LOCATED IN
FUNCS,MAC (SECTION A)

FACOS = COMPUTES ARCCOS IN RADIANS OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION H)

FACOSD = COMPUTES ARCCOS IN DEGREES OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION P)

FASIN = COMPUTES THE ARCSIN IN RADIANS OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION H)

FASIND = COMPUTES ARCSIN IN DEGREES OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION D)

FATAN = COMPUTES THE ARCTAN IN RADIANS OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION L)

FATAND = COMPUTES THE ARCTAN IN DEGREES OF THE NUMBER IN AC=A,
LOCATED IN FUNCS,MAC (SECTION G)

FCOS = COMPUTES THE COSINE OF THE ANGLE (RADIANS) IN AC=A,
LOCATED IN FUNCS,MAC (SECTION J)

FCOSD = COMPUTES THE COSINE OF THE ANGLE (DEGREES) IN AC=A,
LOCATED IN FUNCS,MAC (SECTION H)

FEXP = COMPUTES THE EXPONENTIAL OF THE NUMBER IN AC=A, LOCATED IN
FUNCS,MAC (SECTION E)

FINT = TRUNCATES THE NUMBER IN AC=A TO AN INTEGER, LOCATED IN
FUNCS,MAC (SECTION G)
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>FLOG = computes the common logarithm of the number in AC-A, located in FUNCS,MAC (Section F)</td>
</tr>
<tr>
<td>108</td>
<td></td>
</tr>
<tr>
<td>109</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td></td>
</tr>
<tr>
<td>119</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td></td>
</tr>
<tr>
<td>121</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td></td>
</tr>
<tr>
<td>126</td>
<td></td>
</tr>
<tr>
<td>127</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
</tr>
<tr>
<td>129</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td></td>
</tr>
<tr>
<td>131</td>
<td></td>
</tr>
<tr>
<td>132</td>
<td></td>
</tr>
<tr>
<td>133</td>
<td></td>
</tr>
<tr>
<td>134</td>
<td></td>
</tr>
</tbody>
</table>
160 \textbf{DATA} = STARTING ADDRESS FOR STORAGE OF STUDENT VARIABLES, LESSON VARIABLES, AND STUDENT PERFORMANCE DATA

164 \textbf{DBLK} = CALL STATEMENT STORES DISK BLOCK NO. HERE FOR USE BY RETURN STATEMENT

166 \textbf{DEVICE} = DEVICE NO. ASSOCIATED WITH USER'S TERMINAL

169 \textbf{FIELD} = TOTAL FIELD WIDTH AND NO. OF DIGITS AFTER DECIMAL POINT (USED BY FLOATING OUTPUT ROUTINE) (2 WORDS)

172 \textbf{FRMTOT} = TOTAL NO. OF FRAMES EXECUTED

174 \textbf{IDXSAV} = DATA INDEX IS STORED HERE SO THAT AC-F CAN BE USED

175 \textbf{IDX1} = ADDRESS OF 1ST INDEX VARIABLE

177 \textbf{IDXIV} = FINAL VALUE OF 1ST INDEX

180 \textbf{IDX2} = ADDRESS OF 2ND INDEX VARIABLE

183 \textbf{IDX2V} = INITIAL VALUE OF 2ND INDEX AND FINAL VALUE OF 2ND INDEX (2 WORDS)

187 \textbf{IDX} = INDEX TO THE 1ST WORD CONTAINING STUDENT PERFORMANCE DATA

188 \textbf{INAM} = DISK LOOKUP ARGUMENTS (4 WORDS)

190 \textbf{LESHRO2 (#DIGITS)} = NO. OF SPACES WHICH MUST BE TYPED AFTER LEFT-JUSTIFIED NO.

192 \textbf{LKSAY} = LOOKUP ARGUMENTS FOR ORIGINAL LESSON (4 WORDS)

194 \textbf{NNUM} = NO. OF NEUTRAL ANSWERS

199 \textbf{NUMNT} = INDEX TO NUMBERS STORED IN THE CONSTANT LIST BY AN INPUT STATEMENT

200 \textbf{PC3} = SUBROUTINES STORE RETURN ADDRESS HERE

201 \textbf{PC4} = SUBROUTINES STORE RETURN ADDRESS HERE

202 \textbf{POLLS} = POLISH STRING PUSH DOWN LIST (64 WORDS)

205 \textbf{RANNUM} = INTEGER RANDOM NO. USED BY RANDOM FUNCTION

206 \textbf{RRNUN} = NO. OF CORRECT ANSWERS

209 \textbf{RSYNUM} = TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE

211 \textbf{SANS} = STUDENT ANSWER BUFFER (16 WORDS)
<table>
<thead>
<tr>
<th>CALC</th>
<th>MACRO 47(201)</th>
<th>9810</th>
<th>2-AUG-74 PAGE 1-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>266</td>
<td>000010</td>
<td>H10</td>
<td></td>
</tr>
<tr>
<td>267</td>
<td>000016</td>
<td>OFLAGS*16</td>
<td></td>
</tr>
<tr>
<td>268</td>
<td>000017</td>
<td>Q=17</td>
<td></td>
</tr>
<tr>
<td>269</td>
<td>000000</td>
<td>DSKBLK*STORE</td>
<td></td>
</tr>
<tr>
<td>270</td>
<td>000002</td>
<td>DIGITS<em>ESWRD</em>2</td>
<td></td>
</tr>
<tr>
<td>271</td>
<td>000001</td>
<td>K13=1</td>
<td>FOR 16K CPU, SET K13=8</td>
</tr>
<tr>
<td>272</td>
<td>000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>273</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>274</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>275</td>
<td>000000, 44.07</td>
<td>00 000000*</td>
<td>POINT1: POINT 7,CLOSTR</td>
</tr>
<tr>
<td>276</td>
<td>000001</td>
<td>CRFL: ASC12</td>
<td>/</td>
</tr>
<tr>
<td>277</td>
<td>000001</td>
<td>015 012 000 000 000 /</td>
<td></td>
</tr>
<tr>
<td>278</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>279</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>280</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>281</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>282</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>283</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>284</td>
<td>000002* 132 00 00 000000*</td>
<td>CALCMI: SKIPE RANNUM IF RANNUM IS RANDOM NO. GENERATOR BEEN INITIALIZED</td>
<td></td>
</tr>
<tr>
<td>285</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, *3</td>
<td>YES, JUMP</td>
</tr>
<tr>
<td>286</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: A, [01024.0287]</td>
<td></td>
</tr>
<tr>
<td>287</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: A, RANNUM</td>
<td>ISET INITIAL RANDOM INTEGER TO 01024.0287</td>
</tr>
<tr>
<td>288</td>
<td>000001* 132 00 00 000000*</td>
<td>TLOA OFLAGS*1</td>
<td>ISET COMMAND LEVEL FLAG</td>
</tr>
<tr>
<td>289</td>
<td>000001* 132 00 00 000000*</td>
<td>CALC: TLO OFLAGS*1</td>
<td>CLEAR COMMAND LEVEL FLAG</td>
</tr>
<tr>
<td>290</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: Q,PCS ISAVE RETURN ADDRESS</td>
<td></td>
</tr>
<tr>
<td>291</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: INDEX, IDXXAY ISAVE THE DATA INDEX</td>
<td></td>
</tr>
<tr>
<td>292</td>
<td>000001* 132 00 00 000000*</td>
<td>TLO OFLAGS*2</td>
<td>ISET THE CALC MODE FLAG</td>
</tr>
<tr>
<td>293</td>
<td>000001* 132 00 00 000000*</td>
<td>TCALL 3,[ASC12</td>
<td>/</td>
</tr>
<tr>
<td>294</td>
<td>000001* 132 00 00 000000*</td>
<td>INPUT CHARACTER FROM USER'S TERMINAL</td>
<td></td>
</tr>
<tr>
<td>295</td>
<td>000001* 132 00 00 000000*</td>
<td>TICALL 4,4</td>
<td>INPUT CHARACTER FROM USER'S TERMINAL</td>
</tr>
<tr>
<td>296</td>
<td>000001* 132 00 00 000000*</td>
<td>EAL: A,132</td>
<td></td>
</tr>
<tr>
<td>297</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: A,33</td>
<td></td>
</tr>
<tr>
<td>298</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, *2</td>
<td>IF CHARACTER IS ESC, JUMP</td>
</tr>
<tr>
<td>299</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, PROG, IELSE, GO PROCESS CALC STATEMENT</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: INDEX, IDXXAY IRESTORE DATA INDEX</td>
<td></td>
</tr>
<tr>
<td>301</td>
<td>000001* 132 00 00 000000*</td>
<td>TLO OFLAGS*2</td>
<td>ICLEAR THE CALC MODE FLAG</td>
</tr>
<tr>
<td>302</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, *PC3</td>
<td>RETURN</td>
</tr>
<tr>
<td>303</td>
<td>000001* 132 00 00 000000*</td>
<td>JSP: Q,PCS ICALL THE CALC PROCESSOR IN CPROS,MAC</td>
<td></td>
</tr>
<tr>
<td>304</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, CIN, TIF ERROR, GO TO CIN, TIF</td>
<td></td>
</tr>
<tr>
<td>305</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: Q,POINT1</td>
<td>ELSE PUT BYTE POINTER IN AG=6</td>
</tr>
<tr>
<td>306</td>
<td>000001* 132 00 00 000000*</td>
<td>SET: NUMPNT</td>
<td>ISET NUMBER POINTER TO ZERO</td>
</tr>
<tr>
<td>307</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, PROG, IELSE, GO TO SELECT (SECTION E)</td>
<td></td>
</tr>
<tr>
<td>308</td>
<td>000001* 132 00 00 000000*</td>
<td>JSP Q,PCS</td>
<td>ICALL THE Q-FRAME ADDRESS</td>
</tr>
<tr>
<td>309</td>
<td>000001* 132 00 00 000000*</td>
<td>TLO OFLAGS*2</td>
<td>ICLEAR THE Q-FRAME FLAG</td>
</tr>
<tr>
<td>310</td>
<td>000001* 132 00 00 000000*</td>
<td>TLE OFLAGS*2</td>
<td>ICLEAR THE Q-FRAME FLAG</td>
</tr>
<tr>
<td>311</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: INDEX, IDXXAY ISAVE THE DATA INDEX</td>
<td></td>
</tr>
<tr>
<td>312</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: Q,PCS</td>
<td>ISAVE THE RETURN ADDRESS</td>
</tr>
<tr>
<td>313</td>
<td>000001* 132 00 00 000000*</td>
<td>TLE OFLAGS*2</td>
<td>ISAVE THE RETURN ADDRESS</td>
</tr>
<tr>
<td>314</td>
<td>000001* 132 00 00 000000*</td>
<td>JRSR, *2</td>
<td>YES, JUMP</td>
</tr>
<tr>
<td>315</td>
<td>000001* 132 00 00 000000*</td>
<td>JSP Q,11</td>
<td>ELSE, INPUT &lt;15</td>
</tr>
<tr>
<td>316</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: A,DSKBLK</td>
<td>IGET CURRENT DISK BLOCK NO.</td>
</tr>
<tr>
<td>317</td>
<td>000001* 132 00 00 000000*</td>
<td>MOV: A,DSKAV</td>
<td>ISAVE IT</td>
</tr>
</tbody>
</table>
SECTION B. PROCESSES ITERATION FIELDS.

ITER1: TRE CFLAGS,7 ICLEAR THE ITERATION FLAGS
ITER1: JSP Q,11 IINPUT <02>
ITER1: JSP Q,11 IINPUT 1ST DIGIT OF VARIABLE ADDRESS
MOVE B,-48(A) IPUT VALUE OF DIGIT IN AC-B
INC A,1 IPUT 1 DIGIT LEFT (BASE 09)
JSP Q,11 IPUT 2ND DIGIT OF ADDRESS
ADD B,-48(A) IADD DIGIT TO NO. IN AC-B
MOVE A,RVNUM IPUT NO. OF WORDS OF STUDENT MEMORY IN AC-A
ADD B,DATA(A) IADD ON BASE ADDRESS OF LESSON MEMORY
JRP ITER3 I2ND INDEX VARIABLE?
JRP ITER3 IELSE, SAVE ADDRESS OF INDEX VARIABLE IN IDX1
ITER2: MOVE B,IDX1 ISAVE ADDRESS OF 2ND INDEX VARIABLE IN IDX2
ITER3: MOVE G.POINT1 IPUT POINT IN AC-G
ITER3: JSP Q,11 INPUT CHARACTER
IDPB A,G IPUT IT IN CALC STATEMENT BUFFER
CAIE A,"" END OF EXPRESSION?
JRP 1,3 IO. GO INPUT NEXT CHARACTER
MOVE G.POINT1 IPUT BYTE POINTER
JSP Q,17 IRESET EXPRESSION IN THE BUFFER
MOVE B,POSSL.Move VALUE OF EXPRESSION IN AC-B
TRNE CFLAGS,4 I2ND INDEX?
JRP ITER4 IYES, GO TO ITER4
JRP ITER4 ISET INDEX VARIABLE TO INITIAL VALUE.
CALC MACRO 47(024)-7 081112 8-4UG-74 PAGE 1-12

<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Machine Code</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>637</td>
<td>000475'</td>
<td>254 00 00 0000521</td>
<td>JST JST MEM R: IP NO. WILL BE STORED OUT OF PROPER AREA; 10 TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>638</td>
<td>000476'</td>
<td>202 10 00 000000</td>
<td>MDK2: MOVE H, (P) TO STORE NO. IN PROPER LOCATION</td>
</tr>
<tr>
<td>640</td>
<td>000477</td>
<td>254 00 00 0000433</td>
<td>JST TD2: IGO TO TD2</td>
</tr>
<tr>
<td>641</td>
<td>000520</td>
<td>265 17 00 000000</td>
<td>VID2: IGP, C, INDOR GET ADDRS OF 2ND INDEX VARIABLE</td>
</tr>
<tr>
<td>642</td>
<td>000521</td>
<td>607 16 00 000000</td>
<td>TLNN CFLAGS2: CALC Model</td>
</tr>
<tr>
<td>643</td>
<td>000522</td>
<td>270 03 00 0000463</td>
<td>ADD C,RSVNUM INO ADD ON NO. OF WORDS OF STUDENT DATA</td>
</tr>
<tr>
<td>644</td>
<td>000523</td>
<td>271 03 00 0000474</td>
<td>ADD C, ADDRESS OF DATA AREA</td>
</tr>
<tr>
<td>645</td>
<td>000524</td>
<td>202 01 00 0000446</td>
<td>MOVE A, (C) ISAVE THE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>646</td>
<td>000551</td>
<td>280 01 00 000000</td>
<td>IFN KI10, &lt; FIX C, A CONVERT IT TO AN INTEGER</td>
</tr>
<tr>
<td>648</td>
<td>000557</td>
<td>234 00 00 0000434</td>
<td>MOVE C, A PUT INTEGER IN AC-C</td>
</tr>
<tr>
<td>651</td>
<td>000558</td>
<td>254 00 00 0000477</td>
<td>MEMR EMEMR CALL 3; ASCIZ</td>
</tr>
<tr>
<td>TOKEN</td>
<td>VALUE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAIL A, 60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALE A, 71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT2</td>
<td>IF END OF NO., GO TO MT2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMULI B, 12</td>
<td>SHIFT NO., IN AC-Q ONE DIGIT LEFT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDI B, +68(A)</td>
<td>ADD ON VALUE OF DIGIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST HT5</td>
<td>GO NEXT DIGIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JUMP C, MT4</td>
<td>JUMP IF COLUMN NO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+1</td>
<td>ELSE, SAVE NO. OF ROWS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAIL A, &quot;Y&quot;</td>
<td>ARE THE NO. OF COLUMNS SPECIFIED?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AQJA CM13</td>
<td>YES, GO GET COLUMN NO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, 1</td>
<td>SETE f, NO. OF COLUMNS TO 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+2</td>
<td>ELSE, SET NO. OF COLUMNS TO 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILDB A, G</td>
<td>GET NEXT CHARACTER FROM CALC STATEMENT BUFFER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAIL A, 15</td>
<td>END OF STATEMENT?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT5</td>
<td>YES, GO TO MT5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETE C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT5</td>
<td>GET DIGIT OF FORMAT NO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAIL A, 60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALE A, 71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT6</td>
<td>IF END OF NO., GO TO MT6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMULI B, 12</td>
<td>SHIFT NO., IN AC-Q ONE DIGIT LEFT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDI B, +68(A)</td>
<td>ADD ON VALUE OF DIGIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT7</td>
<td>GO NEXT DIGIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JUMP C, MT9</td>
<td>JUMP IF 2ND FORMAT NO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+1</td>
<td>ELSE, STORE TOTAL FIELD WIDTH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AQJA CM6</td>
<td>YES, GET NO. OF DIGITS AFTER DECIMAL POINT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+4</td>
<td>STORE NO. OF DIGITS AFTER DECIMAL POINT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT9</td>
<td>GO TO MT9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+5</td>
<td>GET NO. OF COLUMNS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+2</td>
<td>PUT NO. IN CONST+2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSGS CONST+1</td>
<td>ANY ROWS LEFT TO TYPE?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST ITERAT</td>
<td>NO. GO TO ITERAT(SECTION D)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT11</td>
<td>YES, GO TO MT11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETH CONST+3</td>
<td>SET NO. OF COLUMNS TO ZERO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+2</td>
<td>PUT NO. OF COLUMNS IN AC-Q</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+5</td>
<td>SAVE NO. IN CONST+5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HR2G CONST+2</td>
<td>ANY COLUMNS LEFT IN THIS ROW?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST RSTCOL</td>
<td>NO. GO TO RESET THE NO. OF COLUMNS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, CONST+3</td>
<td>DECREMENT NO. OF COLUMNS LEFT TO TYPE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSSO CONST+2</td>
<td>ANY COLUMNS LEFT IN THIS ROW?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, FIELD</td>
<td>GET TOTAL FIELD WIDTH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV B, FIELD</td>
<td>PUT IT IN FIELD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV C, CONST+1</td>
<td>GET NO. OF DIGITS AFTER DECIMAL POINT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV C, FIELD+1</td>
<td>PUT IT IN FIELD+1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JUMP B, +2</td>
<td>JUMP IF FIELD WIDTH NOT ZERO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETH FIELD</td>
<td>ELSE, INDICATE STANDARD FORMAT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV A, CONST</td>
<td>GET NO. TO BE_TYPED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADS CONST</td>
<td>ADD ONE TO ARRAY ADDRESS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSPG OUTLP</td>
<td>TYPE THE NO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKIPE FLD</td>
<td>STANDARD FORMAT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSPR +1</td>
<td>YES, JUMP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKIPE CONST+2</td>
<td>END OF LINE?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRST MT12</td>
<td>NO. GO TO MT12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECTION H. EXECUTES IF STATEMENTS.

759 00641 265 17 0 0 001163 1 IFI JSP Q, POLISH IF EVALUATE CONDITIONAL EXPRESSION
760 00647 322 02 0 0 000174 IFM JUMPE B, ITERAT; IF FALSE, GO TO ITERAT (SECTION D)
761 00648 200 02 0 0 000007 MOVE B,0 ELSE, SAVE BYTE POINTER
762 00651 661 16 0 0 040000 TLD CFRLAGS,40960 ISET IF STATEMENT FLAG
763 00652 254 00 0 0 00241 JST SELECT IGO EXECUTE REST OF STATEMENT

SECTION I. EXECUTES GI STATEMENTS.

768 00653 40 02 0 0 00020 SBI SETZ B
780 00654 201 03 0 0 00006 MOVEI C,6 GET DIGIT OF FRAME NO.
771 00655 134 01 0 0 00007 ILDP A,G
772 00656 221 02 0 0 00012 IMUL B,12 SHIFT NO. IN AC-0 ONE DIGIT LEFT
773 00657 271 02 0 0 77772 ADDI B,-60(A) AAD VALUE OF DIGIT TO NO. IN AC-B
774 00660 367 03 0 0 00865 SIMPLE C,-3 IGO GET NEXT DIGIT IF ANY
775 00661 280 08 0 0 000052 MOVE INDEX,idosav IRESTORE THE DATA INDEX
776 00662 294 00 0 0 00020 JST BCOM2 IGO TO BCOM2 IN ACTION.MAC

SECTION J. EXECUTES GI STATEMENTS.

781 00663 220 01 0 0 00000* SGI MOVE A, SUBPDP IGET PUSH DOWN POINTER
782 00664 554 02 0 0 00001 HLRE B,M IGET NO. OF ITEMS IN PUSH DOWN LIST
783 00665 301 02 0 0 00011 CALL B,11 LESS THAN 9 ITEMS
784 00666 254 00 0 0 00674 JST SUBERR IND, GO TYPE ERROR MESSAGE
785 00667 261 05 0 0 00667 PUSH B,0 B,0, BLOCK NO. IN PUSH DOWN LIST
786 00668 261 01 0 0 00014 PUSH A,14 INPUT BUFFER BYTE POINTER IN PUSH DOWN LIST
787 00671 261 01 0 0 00015 PUSH A,15 INPUT BUFFER BYTE COUNT IN PUSH DOWN LIST
788 00672 222 01 0 0 00663 MOVE A, SUBPDP ISAVE THE PUSH DOWN POINTER
789 00673 254 00 0 0 00653 JST SB 1 GO TO SB (SECTION I)

SUBERRS TCALL 3, IASCIE /
SUBROUTINES NESTED MORE THAN 3 LEVELS DEEP (CALC)

904 00675 254 04 0 0 00000 MOVE INDEX,idosav IRESTORE DATA INDEX
| 849 | 000752 |  | 220 07 | 00 000001 | MOVE G.POINT1 | RESET CALC BUFFER BYTE POINTER |
| 850 | 000753 |  | 220 17 | 00 000011 | JSP D.11 | INPUT CHARACTER |
| 851 | 000754 |  | 136 01 | 00 000037 | IDPE A.07 | PUT IT IN CALC STATEMENT BUFFER |
| 852 | 000755 |  | 302 01 | 00 000015 | CAIE A.15 | END OF LINE? |
| 853 | 000756 |  | 254 06 | 00 000053 | JIRST -3 | IND. GO INPUT NEXT CHARACTER |
| 854 | 000757 |  | 220 07 | 00 000000 | MOVE G.POINT1 | RESET BYTE POINTER |
| 855 | 000758 |  | 420 16 | 00 000050 | FL61 TRC CFLAGS,2000 ICLREAD NEGATIVE NO. FLAG |
| 856 | 000759 |  | 220 32 | 00 000007 | ILDB A.B | ISAVE BYTE POINTER IN AC-B |
| 857 | 000760 |  | 134 01 | 00 000022 | CAIE A:"" | GET CHARACTER FROM CALC STATEMENT BUFFER |
| 858 | 000761 |  | 302 01 | 00 000055 | JIRST -3 | IND. JUMP |
| 859 | 000762 |  | 254 07 | 00 000090 | MVC C.102 | ELSE, PUT NEW BYTE POINTER IN AC-G |
| 860 | 000763 |  | 660 16 | 00 000000 | TRC CFLAGS,2000 ISET NEGATIVE NO. FLAG |
| 861 | 000764 |  | 220 17 | 00 000000 | JSP Q1INFLP | "INPUT" NO. |
| 862 | 000765 |  | 622 16 | 00 000000 | TRN CFLAGS,2000 IIS NO. NEGATIVE? |
| 863 | 000766 |  | 213 00 | 00 000022 | MOVNS B1YES, NEGATE IT |
| 864 | 000767 |  | 220 03 | 00 000042 | MOVE A1IDX | GET INITIAL DATA INDEX |
| 865 | 000768 |  | 220 03 | 00 000025 | MOVE C1DATA(C) | PUT 1ST ADDRESS OF STUDENT PERM. DATA IN AC-G |
| 866 | 000769 |  | 317 03 | 00 000026 | CAMG C,CONST | INCL NO. BE STORED IN PROPER AREA? |
| 867 | 000770 |  | 254 08 | 00 000011 | JIRST MEMHDR | IND. GO TYPE ERROR MESSAGE (SECTION F) |
| 868 | 000771 |  | 322 02 | 00 000074 | MOVHE B11CONST | STORE THE NO. |
| 869 | 000772 |  | 350 08 | 00 000072 | AOS CONST | INCREMENT ARRAY ADDRESS |
| 870 | 000773 |  | 370 00 | 00 000031 | SOS CONST1 | DECREMENT NO. OF NUMBERS TO INPUT |
| 871 | 000774 |  | 306 01 | 00 000054 | CAIE A:"" | END OF LINE OF NO.? |
| 872 | 000775 |  | 220 04 | 00 000060 | JIRST FL6 | GO TO FL6 |
| 873 | 000776 |  | 220 04 | 00 000059 | JIRST FL5 | GO TO FL5 |
| 874 | 000777 |  | 220 03 | 00 000055 | STRFILL C1POINT 7,02 | GET 1ST PART OF BYTE POINTER |
| 875 | 000778 |  | 940 03 | 00 000077 | HRR C,CONST | ADD ON THE ADDRESS PORTION OF THE BYTE POINTER |
| 876 | 000779 |  | 134 01 | 00 000037 | ILDB A.B | GET CHARACTER FROM CALC STATEMENT BUFFER |
| 877 | 000780 |  | 306 01 | 00 000042 | CAIE A:"" | END OF STRING? |
| 878 | 000781 |  | 220 00 | 00 000047 | JIRST ITERAT | YES, GO TO ITERAT |
| 879 | 000782 |  | 254 00 | 00 000023 | IDC A."" | IND. GO TO AC-C |
| 880 | 000783 |  | 220 00 | 00 000092 | MOVE B1IDX | ELSE, STORE THE CHARACTER |
| 881 | 000784 |  | 220 02 | 00 000073 | MOVE B1DATA(B) | GET INITIAL DATA INDEX |
| 882 | 000785 |  | 201 02 | 00 000035 | CAIE B("C") | WAS CHARACTER STORED IN DATA AREA? |
| 883 | 000786 |  | 307 02 | 00 000000 | JIRST MEMHDR | IND. GO TYPE ERROR MESSAGE |
| 884 | 000787 |  | 220 00 | 00 000006 | JIRST -18 | IND. GET NEXT CHARACTER |

**SECTION H. EXECUTES TYPE STATEMENTS:**

<p>| 885 | 000788 |  | 100 00 | 00 000000 | CTYPE | MOVE G.G | PUT BYTE POINTER IN AC-G |
| 886 | 000789 |  | 134 01 | 00 000007 | ILDB A.G | GET CHARACTER FROM CALC STATEMENT BUFFER |
| 888 | 000790 |  | 306 01 | 00 000054 | CAIE A:&quot;&quot; | IND. CHARACTER A COMMA? |
| 889 | 000791 |  | 220 00 | 00 000017 | JIRST CTYPE | YES. GO INPUT NEXT CHARACTER |
| 892 | 000792 |  | 306 01 | 00 000042 | GTPI | CAIE A.&quot;&quot; | IND. CHARACTER A QUOTE MARK? |
| 893 | 000793 |  | 220 04 | 00 000046 | JIRST TYPXT | YES. GO TO TYPXT |
| 894 | 000794 |  | 302 01 | 00 000017 | CAIE A:&quot;&quot; | END STATEMENT OR TEXT_EXPRESSION, GO TO ITERAT (SEC. D) |
| 895 | 000795 |  | 306 01 | 00 000015 | CAIE A:&quot;&quot; | END STATEMENT OR TEXT_EXPRESSION, GO TO ITERAT (SEC. D) |
| 896 | 000796 |  | 220 07 | 00 000000 | MOVE G.G | ELSE, PUT BYTE POINTER IN AC-G |</p>
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>000000</td>
<td>011000101</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1001</td>
<td>000000</td>
<td>011000100</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
<tr>
<td>1002</td>
<td>000000</td>
<td>011000110</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1003</td>
<td>000000</td>
<td>011000111</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
<tr>
<td>1004</td>
<td>000000</td>
<td>011001000</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1005</td>
<td>000000</td>
<td>011001001</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
<tr>
<td>1006</td>
<td>000000</td>
<td>011001010</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1007</td>
<td>000000</td>
<td>011001011</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
<tr>
<td>1008</td>
<td>000000</td>
<td>011001100</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1009</td>
<td>000000</td>
<td>011001101</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
<tr>
<td>1010</td>
<td>000000</td>
<td>011001110</td>
<td>MOVE IN I-ADDRESS</td>
</tr>
<tr>
<td>1011</td>
<td>000000</td>
<td>011001111</td>
<td>MOVE OUT I-ADDRESS</td>
</tr>
</tbody>
</table>

**Notes:**
- The constant list is placed in the push down list at the completion of this process.
- The address of the value of the Polish expression is at the top of the push down list.
- The program includes instructions for moving values in and out of the address space, as well as pushing and popping items from a stack.
<table>
<thead>
<tr>
<th></th>
<th>ONEI</th>
<th>581#</th>
<th></th>
<th>OPIN</th>
<th>1018#</th>
<th>1102#</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OUTFLP</td>
<td>254#</td>
<td></td>
<td>PATH</td>
<td>1035#</td>
<td>1335#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>POL</td>
<td>354#</td>
<td></td>
<td>POL1</td>
<td>1015#</td>
<td>1056#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PC1</td>
<td>290#</td>
<td></td>
<td>POS</td>
<td>1062#</td>
<td>1220#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PC2</td>
<td>1013#</td>
<td></td>
<td>POLL</td>
<td>1062#</td>
<td>1297#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PC3</td>
<td>313#</td>
<td></td>
<td>POLISH</td>
<td>379#</td>
<td>799#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>POLLST</td>
<td>295#</td>
<td></td>
<td>POLSEX</td>
<td>1557#</td>
<td>1539#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPI</td>
<td>1702#</td>
<td></td>
<td>PP2</td>
<td>1704#</td>
<td>1903#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PP3</td>
<td>1633#</td>
<td></td>
<td>PP4</td>
<td>1704#</td>
<td>1703#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PROS</td>
<td>385#</td>
<td></td>
<td>PTH1</td>
<td>1356#</td>
<td>1359#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PTH2</td>
<td>1355#</td>
<td></td>
<td>PTH3</td>
<td>1554#</td>
<td>1375#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PTH4</td>
<td>1354#</td>
<td></td>
<td>PTH5</td>
<td>1354#</td>
<td>1375#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PTH6</td>
<td>1354#</td>
<td></td>
<td>PTH7</td>
<td>1354#</td>
<td>1375#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PTHX</td>
<td>1354#</td>
<td></td>
<td>PUSH</td>
<td>1049#</td>
<td>1901#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R101</td>
<td>1035#</td>
<td></td>
<td>R201</td>
<td>1035#</td>
<td>1056#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISE</td>
<td>1526#</td>
<td></td>
<td>RANNUM</td>
<td>249#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>REPLACE</td>
<td>249#</td>
<td>455#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISEO</td>
<td>1546#</td>
<td></td>
<td>RESET</td>
<td>1183#</td>
<td>1183#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOVO</td>
<td>1546#</td>
<td></td>
<td>RETSUB</td>
<td>1183#</td>
<td>249#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISEOVO</td>
<td>1546#</td>
<td></td>
<td>RETURN</td>
<td>1183#</td>
<td>1183#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISEOVO</td>
<td>1546#</td>
<td></td>
<td>R101</td>
<td>259#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISEOVO</td>
<td>1546#</td>
<td></td>
<td>R201</td>
<td>259#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISEOVO</td>
<td>1546#</td>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOVO</td>
<td>1546#</td>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td>284#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R101</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R201</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R301</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R401</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R501</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R601</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R701</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R801</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R901</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R101</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R111</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R121</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R131</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R141</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAISOV</td>
<td>1546#</td>
<td></td>
<td>R151</td>
<td>1254#</td>
<td>1292#</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The table above contains code or data entries, possibly representing a computer program or a set of instructions.
<table>
<thead>
<tr>
<th>SG</th>
<th>493</th>
<th>782#</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCR</td>
<td>494</td>
<td>795#</td>
</tr>
<tr>
<td>SIN</td>
<td>1143</td>
<td>1237#</td>
</tr>
<tr>
<td>SIND</td>
<td>1144</td>
<td>1241#</td>
</tr>
<tr>
<td>SINDX</td>
<td>1146</td>
<td>1241#</td>
</tr>
<tr>
<td>SN1</td>
<td>1326</td>
<td>1330#</td>
</tr>
<tr>
<td>SN3</td>
<td>1326</td>
<td>1332#</td>
</tr>
<tr>
<td>SNAN</td>
<td>1328</td>
<td>1352#</td>
</tr>
<tr>
<td>SPCVAR</td>
<td>1326</td>
<td>1330#</td>
</tr>
<tr>
<td>SORT</td>
<td>1134</td>
<td>1169#</td>
</tr>
<tr>
<td>SSS</td>
<td>1043</td>
<td>1059#</td>
</tr>
<tr>
<td>STA1</td>
<td>1430</td>
<td>1452#</td>
</tr>
<tr>
<td>STA2</td>
<td>1432</td>
<td>1462#</td>
</tr>
<tr>
<td>STORE</td>
<td>934</td>
<td>929#</td>
</tr>
<tr>
<td>STRFIL</td>
<td>834</td>
<td>875#</td>
</tr>
<tr>
<td>SUBERR</td>
<td>789</td>
<td>791#</td>
</tr>
<tr>
<td>SUBDV</td>
<td>1491</td>
<td>1493#</td>
</tr>
<tr>
<td>SUBDLP</td>
<td>252</td>
<td>782#</td>
</tr>
<tr>
<td>SBT</td>
<td>1106</td>
<td>1485#</td>
</tr>
<tr>
<td>TAG</td>
<td>1459</td>
<td>1400#</td>
</tr>
<tr>
<td>TAG1</td>
<td>1405</td>
<td>1406#</td>
</tr>
<tr>
<td>TAG2</td>
<td>1417</td>
<td>1422#</td>
</tr>
<tr>
<td>TAN</td>
<td>1141</td>
<td>1216#</td>
</tr>
<tr>
<td>TANO</td>
<td>1142</td>
<td>1206#</td>
</tr>
<tr>
<td>TOL</td>
<td>622</td>
<td>653#</td>
</tr>
<tr>
<td>TIME</td>
<td>1020</td>
<td>1053#</td>
</tr>
<tr>
<td>TNUM</td>
<td>292</td>
<td>1046</td>
</tr>
<tr>
<td>TOL</td>
<td>1251</td>
<td>1256</td>
</tr>
<tr>
<td>TTT</td>
<td>1034</td>
<td>1054#</td>
</tr>
<tr>
<td>TTYIN</td>
<td>42</td>
<td>513#</td>
</tr>
<tr>
<td>TWD</td>
<td>42</td>
<td>438#</td>
</tr>
<tr>
<td>TWOD</td>
<td>582</td>
<td>616#</td>
</tr>
<tr>
<td>TYPXT</td>
<td>699</td>
<td>941#</td>
</tr>
<tr>
<td>UUU</td>
<td>1039</td>
<td>1085#</td>
</tr>
<tr>
<td>VARIN</td>
<td>1032</td>
<td>1085#</td>
</tr>
<tr>
<td>VIX</td>
<td>978</td>
<td>602#</td>
</tr>
<tr>
<td>VIXD</td>
<td>620</td>
<td>641#</td>
</tr>
<tr>
<td>WNIN</td>
<td>252</td>
<td>1851#</td>
</tr>
<tr>
<td>WRTNG</td>
<td>1154</td>
<td>1337#</td>
</tr>
<tr>
<td>WW</td>
<td>1038</td>
<td>1081#</td>
</tr>
<tr>
<td>ZERD</td>
<td>1513</td>
<td>1521#</td>
</tr>
<tr>
<td>BESEX</td>
<td>1533</td>
<td>1553#</td>
</tr>
<tr>
<td>KNX</td>
<td>1041</td>
<td>1056#</td>
</tr>
</tbody>
</table>
TITLE ACTION

COMMENT

WRITTEN BY GARRET A. VANDER LUGT COMPLETED 18-MAY-74

PART OF LEARN.MAC, VERSION 3

ACTION.MAC EXECUTES THE ACTIONS FOLLOWING ACTION STATEMENTS IN G-FRAMES.

CONTENTS OF ACTION.MAC

SECTION A, EXECUTES F1 STATEMENTS.
SECTION B, SUBROUTINE WHICH TYPES RANDOM COMMENTS.
SECTION C, EXECUTES C1 STATEMENTS.
SECTION D, EXECUTES R1 STATEMENTS.
SECTION E, EXECUTES A1 STATEMENTS.
SECTION F, EXECUTES G1 STATEMENTS.
SECTION G, EXECUTES S1 STATEMENTS.
SECTION H, EXECUTES E1 STATEMENTS.
SECTION I, EXECUTES Li STATEMENTS.
SECTION J, EXECUTES G1 STATEMENTS.
SECTION K, EXECUTES B1 STATEMENTS.
SECTION L, EXECUTES P1 STATEMENTS.

ENTRY POINTS TO ACTION.MAC

ACMD = FROM INTERP.MAC (SECTION 21), LOCATED IN SECTION E
BCMD = FROM INTERP.MAC (SECTION 21), LOCATED IN SECTION K
BCMD2 = FROM INTERP.MAC (SECTION FF) AND CALC.MAC (SECTION I), LOCATED IN SECTION K
CCMD = FROM INTERP.MAC (SECTION 21), LOCATED IN SECTION C
ECMD = FROM INTERP.MAC (SECTION 21), LOCATED IN SECTION M.
<table>
<thead>
<tr>
<th>ACTION MACRO 47(204) 708512 2-AUG-74 PAGE 1-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION MAC 17-MAY-74</td>
</tr>
<tr>
<td>FCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION A</td>
</tr>
<tr>
<td>GCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION J</td>
</tr>
<tr>
<td>LCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION I</td>
</tr>
<tr>
<td>PCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION L</td>
</tr>
<tr>
<td>QCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION F</td>
</tr>
<tr>
<td>RCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION D</td>
</tr>
<tr>
<td>RETSUB = FROM INTERP,MAC (SECTION 2) AND CALC,MAC (SECTION J), LOCATED IN SECTION J</td>
</tr>
<tr>
<td>SCO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION E</td>
</tr>
<tr>
<td>UCCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION G</td>
</tr>
<tr>
<td>UFCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION A</td>
</tr>
<tr>
<td>UPCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION L</td>
</tr>
<tr>
<td>USCHO = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION G</td>
</tr>
<tr>
<td>EXTERNAL SUBROUTINES CALLED BY ACTION,MAC.</td>
</tr>
<tr>
<td>GFRAME = EXECUTES G-FRAME STATEMENTS, LOCATED IN CALC,MAC (SECTION A)</td>
</tr>
<tr>
<td>EXPOR = EXPANDS THE LOW SEGMENT IF NECESSARY, LOCATED IN AUX1,MAC (SECTION M)</td>
</tr>
<tr>
<td>IOD = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1,MAC (SECTION E)</td>
</tr>
<tr>
<td>OUTFLP = OUTPUTS THE FLOATING POINT NO. IN AC-A, LOCATED IN AUX1,MAC (SECTION G)</td>
</tr>
<tr>
<td>PAG = PAGES CRT TERMINALS, LOCATED IN INTERP,MAC (SECTION F)</td>
</tr>
<tr>
<td>POLISH = EVALUATES POLISH EXPRESSIONS, LOCATED IN CALC,MAC (SECTION G)</td>
</tr>
<tr>
<td>SKPTXT = SKIPS CHARACTERS IN A TEXT STATEMENT, LOCATED IN INTERP,MAC (SECTION AA)</td>
</tr>
<tr>
<td>SLDRTN = OPERATES THE SLIDE PROJECTOR INTERFACE, LOCATED IN INTERP,MAC (SECTION F)</td>
</tr>
<tr>
<td>EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOW1,MAC)</td>
</tr>
<tr>
<td>Block</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Block = Disk Block No., Buffer Byte Pointer, and Buffer Byte Count at Beginning of Last Answer Statement (3 Words)</td>
</tr>
<tr>
<td>Common = Random No., Used to Select Random Comments</td>
</tr>
<tr>
<td>Data = Starting Address for Storage of Student Variable, Lesson Variables, and Student Performance Data</td>
</tr>
<tr>
<td>Field = Total Field Width (Used by Floating Output Routine)</td>
</tr>
<tr>
<td>Frame = Current Frame No.</td>
</tr>
<tr>
<td>FHTOT = Total No. of Frames Executed</td>
</tr>
<tr>
<td>FHTYP = Frame Type of Current Frame</td>
</tr>
<tr>
<td>IBuf = Disk Input Buffer Header (3 Words)</td>
</tr>
<tr>
<td>IDXSAV = Data Index Is Saved Here When AC-F Is Used by a Subprogram Other Than Interp, MAC</td>
</tr>
<tr>
<td>Pollst = Polish String Push Down List (64 Words)</td>
</tr>
<tr>
<td>RGTANS = Correct Answer Buffer (16 Words)</td>
</tr>
<tr>
<td>SRORE (DSKBLK) = Current Disk Block No.</td>
</tr>
<tr>
<td>Subdcp = Push Down List for Storing Information Needed to Return from a Subroutine - 1st Word Contains Push Down Pointer (16 Words)</td>
</tr>
<tr>
<td>SVPX = Information Needed to Repeat Question (Q) - Disk Block No., Buffer Byte Pointer, and Buffer Byte Count (3 Words)</td>
</tr>
<tr>
<td>Tag = Tag of the Last Answer</td>
</tr>
</tbody>
</table>

**Description of Flags Stored in AC-8**

- Bit 3 = Correct Answer
- Bit 4 = Wrong Answer
- Bit 28 = Review in Progress

**Description of Flags Stored in AC-16**

- Bit 2 = Q-Frame
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION C. INTERPRETS AND EXECUTES CALC (0) STATEMENTS,
SECTION E. INTERPRETS AND EXECUTES ANSWER (AI) STATEMENTS.

ACMD1 JSP Q,11 INPUT CHARACTER
GAIN A,12 END OF LINE
JSTR NOTXT YES, GO TO NOTXT
TPTXT TEXT

TTCALL CRLF
VSP Q,11
JMP A,*3
TICAL A
JSTR .5
VSP Q,11
TICAL A
JSTR .4
TLO 16,10000
JSTR .11
ACMD2 GO TO ACMD2
NOTXTI TTCALL ASCII /ANSI /
ACMD2 TLINE FLAGS,10 IS STORED ANSWER NUMERIC?
JSTRNM A,12 YES, GO TO NMACMD
MOVE B,PONT13 ELSE, PUT ANSWER POINTER IN AC-B
ILDB A,B GET ANSWER CHARACTER
TICAL A,1 TYPE THE CHARACTER
CAIC A,15 END OF ANSWER
JSTR .3 INC, GO NEXT CHARACTER
TICAL 3,CRLF
ACMD2 GETACT GO TO GETACT IN INTERP.MAC
MOVE G,PONT13 PUT ANSWER POINTER IN AC-G
MOVEM INDEX, IDXSAV ISAVE THE DATA INDEX
USP Q,Polish EVALUATE ANSWER
ASS FLAGE,2 ICLEAR UNITS FLAG
CAIC A,15 END OF ANSWER?
TRO FLAGE,1 YES, SET FLAG
MOVE A,PIOLST PUT ANSWER IN AC-A
MOVEL B,1 SUBMIT THE DATA
MOVEL 6,1 FIELD SET FIELD WIDTH TO 1
SECTION F. INTERPRETS AND EXECUTES QU STATEMENTS.

QCMD1: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD2: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD3: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD4: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD5: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD6: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

SECTION F. INTERPRETS AND EXECUTES QU STATEMENTS.

QCMD1: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD2: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD3: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD4: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD5: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD6: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

SECTION F. INTERPRETS AND EXECUTES QU STATEMENTS.

QCMD1: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD2: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD3: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD4: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD5: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT

QCMD6: JSP Q.11 INPUT CHARACTER
CAIN A.15 END OF LINE?
JRTST STOCH YES, GO TO STOCH
TPRTX 4 TYPE TEXT
<table>
<thead>
<tr>
<th>ACTION MACRO</th>
<th>4712847</th>
<th>00112</th>
<th>2-AUG-74 PAGE 1-10</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Line</th>
<th>Action</th>
<th>Address</th>
<th>Register1</th>
<th>Register2</th>
<th>Immediate1</th>
<th>Immediate2</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>529</td>
<td>MOVEM A,14</td>
<td>002416</td>
<td>202 01</td>
<td>0 0 000014</td>
<td>MOVE A,SPOST=2</td>
<td>MOVEM A,15</td>
<td>SET POINTER TO BEGINNING OF FRAME</td>
</tr>
<tr>
<td>530</td>
<td>MOVEM A,15</td>
<td>002417</td>
<td>200 01</td>
<td>0 0 000020</td>
<td>MOVEM A,15</td>
<td>IPUT BYTE COUNT IN AC-15</td>
<td></td>
</tr>
<tr>
<td>531</td>
<td>MOVEM A,15</td>
<td>002428</td>
<td>128 01</td>
<td>0 0 000015</td>
<td>MOVEM A,15</td>
<td>INCREMENT DATA INDEX</td>
<td></td>
</tr>
<tr>
<td>532</td>
<td>JSP Q,EXPCTR</td>
<td>002429</td>
<td>350 00</td>
<td>0 0 000026</td>
<td>JSP Q,EXPCTR</td>
<td>EXPAND CORE IF NECESSARY</td>
<td></td>
</tr>
<tr>
<td>533</td>
<td>JSP Q,EXPCTR</td>
<td>002430</td>
<td>265 17</td>
<td>0 0 002261</td>
<td>JSP Q,EXPCTR</td>
<td>ADD TO FRAME COUNT</td>
<td></td>
</tr>
<tr>
<td>534</td>
<td>JSP Q,EXPCTR</td>
<td>002431</td>
<td>250 00</td>
<td>0 0 000252</td>
<td>JSP Q,EXPCTR</td>
<td>ADD DUE TO FRAME COUNT</td>
<td></td>
</tr>
<tr>
<td>535</td>
<td>MOVEM A,FRMNHM</td>
<td>002432</td>
<td>200 01</td>
<td>0 0 000253</td>
<td>MOVEM A,FRMNHM</td>
<td>SET CURRENT FRAME NO.</td>
<td></td>
</tr>
<tr>
<td>536</td>
<td>MOVEM A,FRNTHM</td>
<td>002433</td>
<td>600 00</td>
<td>0 0 000280</td>
<td>MOVEM A,FRNTHM</td>
<td>SET CURRENT FRAME TYPE</td>
<td></td>
</tr>
<tr>
<td>537</td>
<td>MOVEM A,FRNTHM</td>
<td>002434</td>
<td>600 00</td>
<td>0 0 000750</td>
<td>MOVEM A,FRNTHM</td>
<td>SET FRAME NO.</td>
<td></td>
</tr>
<tr>
<td>538</td>
<td>MOVEM A,FRNTHM</td>
<td>002435</td>
<td>137 01</td>
<td>0 0 003001</td>
<td>MOVEM A,FRNTHM</td>
<td>SET FRAME NO. TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>539</td>
<td>MOVEM A,FRNTHM</td>
<td>002436</td>
<td>247 01</td>
<td>0 0 000023</td>
<td>MOVEM A,FRNTHM</td>
<td>SET TIME A.</td>
<td></td>
</tr>
<tr>
<td>540</td>
<td>MOVEM A,FRNTHM</td>
<td>002437</td>
<td>231 01</td>
<td>0 0 01750</td>
<td>MOVEM A,FRNTHM</td>
<td>IDIV A,100000</td>
<td>CONVERT TO SECONDS</td>
</tr>
<tr>
<td>541</td>
<td>MOVEM A,FRNTHM</td>
<td>002438</td>
<td>542 01</td>
<td>0 0 000275</td>
<td>MOVEM A,FRNTHM</td>
<td>WARM A,DATA (INDEX), IADD TIME TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>542</td>
<td>MOVEM A,FRNTHM</td>
<td>002439</td>
<td>200 01</td>
<td>0 0 000272</td>
<td>MOVEM A,FRNTHM</td>
<td>WARM A,DATA (INDEX), IADD TIME TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>543</td>
<td>MOVEM A,FRNTHM</td>
<td>002440</td>
<td>137 01</td>
<td>0 0 003002</td>
<td>MOVEM A,FRNTHM</td>
<td>WARM A,DATA (INDEX), IADD TIME TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>544</td>
<td>MOVEM A,FRNTHM</td>
<td>002441</td>
<td>350 00</td>
<td>0 0 000006</td>
<td>MOVEM A,FRNTHM</td>
<td>WARM A,DATA (INDEX), IADD TIME TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>545</td>
<td>MOVEM A,FRNTHM</td>
<td>002442</td>
<td>422 00</td>
<td>0 0 00432</td>
<td>MOVEM A,FRNTHM</td>
<td>SETM A,DATA (INDEX), IADD TIME TO STUD. PERF. DATA</td>
<td></td>
</tr>
<tr>
<td>546</td>
<td>MOVEM A,FRNTHM</td>
<td>002443</td>
<td>265 17</td>
<td>0 0 000511</td>
<td>MOVEM A,FRNTHM</td>
<td>JSP Q,11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>547</td>
<td>MOVEM A,FRNTHM</td>
<td>002444</td>
<td>265 17</td>
<td>0 0 000515</td>
<td>MOVEM A,FRNTHM</td>
<td>CALE A,15</td>
<td>END OF LST LINE IN FRAME?</td>
</tr>
<tr>
<td>548</td>
<td>MOVEM A,FRNTHM</td>
<td>002445</td>
<td>254 00</td>
<td>0 0 00437</td>
<td>MOVEM A,FRNTHM</td>
<td>JRST -2</td>
<td>IND. INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>549</td>
<td>MOVEM A,FRNTHM</td>
<td>002446</td>
<td>254 00</td>
<td>0 0 000000</td>
<td>MOVEM A,FRNTHM</td>
<td>JRST QFRAM1</td>
<td>GO TO QFRAM1 IN INTERP,MAC</td>
</tr>
</tbody>
</table>

**SECTION G. INTERPRETS AND EXECUTES SLIDE (S1) STATEMENTS.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Action</th>
<th>Address</th>
<th>Register1</th>
<th>Register2</th>
<th>Immediate1</th>
<th>Immediate2</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>534</td>
<td>USCDH1</td>
<td>002447</td>
<td>265 17</td>
<td>0 0 000000</td>
<td>USCDH1</td>
<td>JSP Q,SLRTHN</td>
<td>INTERPRET SLIDE STATEMENT</td>
</tr>
<tr>
<td>535</td>
<td>USCDH1</td>
<td>002448</td>
<td>254 00</td>
<td>0 0 00214</td>
<td>USCDH1</td>
<td>JRST UNCAC1</td>
<td>GO TO UNCAC1 IN INTERP,MAC</td>
</tr>
<tr>
<td>536</td>
<td>USCDH1</td>
<td>002449</td>
<td>265 17</td>
<td>0 0 000443</td>
<td>USCDH1</td>
<td>SCHD1</td>
<td>JSP Q,SLRTHN</td>
</tr>
<tr>
<td>537</td>
<td>USCDH1</td>
<td>002450</td>
<td>254 00</td>
<td>0 0 00356</td>
<td>USCDH1</td>
<td>JRST GETAC1</td>
<td>GO TO GETAC1 IN INTERP,MAC</td>
</tr>
</tbody>
</table>

**SECTION H. INTERPRETS AND EXECUTES EXIT (E1) STATEMENTS.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Action</th>
<th>Address</th>
<th>Register1</th>
<th>Register2</th>
<th>Immediate1</th>
<th>Immediate2</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>542</td>
<td>ECDH1</td>
<td>002451</td>
<td>265 17</td>
<td>0 0 000311</td>
<td>ECDH1</td>
<td>JSP Q,11</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>543</td>
<td>ECDH1</td>
<td>002452</td>
<td>306 01</td>
<td>0 0 000015</td>
<td>ECDH1</td>
<td>CAIN A,15</td>
<td>END OF LINE?</td>
</tr>
<tr>
<td>544</td>
<td>ECDH1</td>
<td>002453</td>
<td>254 02</td>
<td>0 0 000020</td>
<td>ECDH1</td>
<td>JRST EOL</td>
<td>YES, GO TO EOL IN INTERP,MAC</td>
</tr>
<tr>
<td>545</td>
<td>ECDH1</td>
<td>002454</td>
<td>251 03</td>
<td>0 0 000000</td>
<td>ECDH1</td>
<td>TYPTXT</td>
<td>TYPE TEXT</td>
</tr>
<tr>
<td>546</td>
<td>ECDH1</td>
<td>002455</td>
<td>265 17</td>
<td>0 0 000011</td>
<td>ECDH1</td>
<td>TCALL 3,GRFD</td>
<td>JSP Q,11</td>
</tr>
<tr>
<td>547</td>
<td>ECDH1</td>
<td>002456</td>
<td>322 01</td>
<td>0 0 00457</td>
<td>ECDH1</td>
<td>TCALL 1, A</td>
<td>JUMPE A,1, A</td>
</tr>
<tr>
<td>548</td>
<td>ECDH1</td>
<td>002457</td>
<td>201 01</td>
<td>0 0 003021</td>
<td>ECDH1</td>
<td>TCALL 1, A</td>
<td>JUMPE A,1, A</td>
</tr>
<tr>
<td>549</td>
<td>ECDH1</td>
<td>002458</td>
<td>254 02</td>
<td>0 0 00453</td>
<td>ECDH1</td>
<td>JRST -3</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td>ECDH1</td>
<td>002459</td>
<td>265 17</td>
<td>0 0 003011</td>
<td>ECDH1</td>
<td>JRST -3</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td>ECDH1</td>
<td>002460</td>
<td>256 01</td>
<td>0 0 003015</td>
<td>ECDH1</td>
<td>CAIN A,15</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td>ECDH1</td>
<td>002461</td>
<td>254 02</td>
<td>0 0 00455</td>
<td>ECDH1</td>
<td>JRST -4</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td>ECDH1</td>
<td>002462</td>
<td>661 16</td>
<td>0 0 100000</td>
<td>ECDH1</td>
<td>TLO 16,100000</td>
<td></td>
</tr>
<tr>
<td>554</td>
<td>ECDH1</td>
<td>002463</td>
<td>265 17</td>
<td>0 0 00001</td>
<td>ECDH1</td>
<td>JSP Q,GFRAME=1</td>
<td></td>
</tr>
<tr>
<td>555</td>
<td>ECDH1</td>
<td>002464</td>
<td>254 02</td>
<td>0 0 00453</td>
<td>ECDH1</td>
<td>JRST -4</td>
<td></td>
</tr>
<tr>
<td>556</td>
<td>ECDH1</td>
<td>002465</td>
<td>254 02</td>
<td>0 0 00001</td>
<td>ECDH1</td>
<td>JRST EOL</td>
<td>GO TO EOL IN INTERP,MAC</td>
</tr>
</tbody>
</table>

**SECTION I. INTERPRETS AND EXECUTES LOGOUT (L1) STATEMENTS.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Action</th>
<th>Address</th>
<th>Register1</th>
<th>Register2</th>
<th>Immediate1</th>
<th>Immediate2</th>
<th>Description</th>
</tr>
</thead>
</table>

---

[Note: The image contains a page from a document with machine code instructions and comments. The instructions are written in hexadecimal format, and the comments are likely in English, indicating the function and purpose of each instruction. The page is part of a larger set of instructions that are used in a computer program or system.]
SECTION J. INTERPRETS AND EXECUTES SUBROUTINE CALLS (G1).

567 0025567 269 01 00 000000  GCMD: MOVE A, SUBDP B GET PUSH DOWN POINTER
568 0025567 269 02 00 000001  HLRZ B A
569 0025567 269 02 00 000001  CAIL B 010 ITEMS IN PUSH DOWN LIST
570 0025567 269 04 00 000001  JSP G 016 SUBERR YES, GO TYPE ERROR MESSAGE
571 0025567 269 01 00 000014  PUSH A 0 DSKBLK SAVE CURRENT DISK BLOCK NO.
572 0025567 269 01 00 000014  PUSH A 14 SAVE CURRENT BUFFER BYTE POINTER
573 0025567 269 01 00 000015  PUSH A 15 SAVE CURRENT BYTE COUNT
574 0025567 269 01 00 000015  MOVEM A, SUBDP B ISAVE PUSH DOWN POINTER
575 0025567 269 00 00 000015  RETSUBI MOVE A, SUBDP B GET PUSH DOWN POINTER
576 0025567 269 00 00 000015  HLRZ B A
577 0025567 269 02 00 000014  JUMPE B, SUBERR G0 TYPE ERROR MESSAGE IF LIST IS EMPTY
578 0025567 269 01 00 000015  POP A 14 GET BYTE POINTER
579 0025567 269 01 00 000015  POP A 14 GET BYTE POINTER
580 0025567 269 01 00 000015  POP A, DSKBLK GET DISK BLOCK NO.
581 0025567 269 01 00 000015  SETE 1, DSKBLK
582 0025567 269 01 00 000015  INPUT 1, INPUT DISK BLOCK
583 0025567 269 01 00 000015  MOVEM A, SUBDP B ISAVE PUSH DOWN POINTER
584 0025567 269 01 00 000015  GTCHR JSP G 016 GET CHARACTER
585 0025567 269 01 00 000015  JSP G 016 IS CHARACTER A 001?
586 0025567 269 01 00 000015  JRST * 3 NO, JUMP
587 0025567 269 01 00 000015  JSP G, SKIP TEXT ISKIP TEXT
588 0025567 269 01 00 000015  JRST GTCHR G0 INPUT NEXT CHARACTER
589 0025567 269 01 00 000015  CAIN A 001 IS BEGINNING OF MESSAGE?
590 0025567 269 01 00 000015  JRST FSSTLINE YES, GO TO FSSTLINE IN INTERP.MAC
591 0025567 269 01 00 000015  CAIN A 04 END OF LESSON?
592 0025567 269 01 00 000015  JRST EOL YES, GO TO EOL IN INTERP.MAC
593 0025567 269 01 00 000015  JRST GTCHR ELSE, GO INPUT NEXT CHARACTER

567 0025567 269 01 00 000000  GCMD: MOVE A, SUBDP B GET PUSH DOWN POINTER
568 0025567 269 02 00 000001  HLRZ B A
569 0025567 269 02 00 000001  CAIL B 010 ITEMS IN PUSH DOWN LIST
570 0025567 269 04 00 000001  JSP G 016 SUBERR YES, GO TYPE ERROR MESSAGE
571 0025567 269 01 00 000014  PUSH A 0 DSKBLK SAVE CURRENT DISK BLOCK NO.
572 0025567 269 01 00 000014  PUSH A 14 SAVE CURRENT BUFFER BYTE POINTER
573 0025567 269 01 00 000015  PUSH A 15 SAVE CURRENT BYTE COUNT
574 0025567 269 01 00 000015  MOVEM A, SUBDP B ISAVE PUSH DOWN POINTER
575 0025567 269 00 00 000015  RETSUBI MOVE A, SUBDP B GET PUSH DOWN POINTER
576 0025567 269 00 00 000015  HLRZ B A
577 0025567 269 02 00 000014  JUMPE B, SUBERR G0 TYPE ERROR MESSAGE IF LIST IS EMPTY
578 0025567 269 01 00 000015  POP A 14 GET BYTE POINTER
579 0025567 269 01 00 000015  POP A 14 GET BYTE POINTER
580 0025567 269 01 00 000015  POP A, DSKBLK GET DISK BLOCK NO.
581 0025567 269 01 00 000015  SETE 1, DSKBLK
582 0025567 269 01 00 000015  INPUT 1, INPUT DISK BLOCK
583 0025567 269 01 00 000015  MOVEM A, SUBDP B ISAVE PUSH DOWN POINTER
584 0025567 269 01 00 000015  GTCHR JSP G 016 GET CHARACTER
585 0025567 269 01 00 000015  JSP G 016 IS CHARACTER A 001?
586 0025567 269 01 00 000015  JRST * 3 NO, JUMP
587 0025567 269 01 00 000015  JSP G, SKIP TEXT ISKIP TEXT
588 0025567 269 01 00 000015  JRST GTCHR G0 INPUT NEXT CHARACTER
589 0025567 269 01 00 000015  CAIN A 001 IS BEGINNING OF MESSAGE?
590 0025567 269 01 00 000015  JRST FSSTLINE YES, GO TO FSSTLINE IN INTERP.MAC
591 0025567 269 01 00 000015  CAIN A 04 END OF LESSON?
592 0025567 269 01 00 000015  JRST EOL YES, GO TO EOL IN INTERP.MAC
593 0025567 269 01 00 000015  JRST GTCHR ELSE, GO INPUT NEXT CHARACTER
SECTION K. INTERPRETS AND EXECUTES BRANCHING (BI) STATEMENTS.

624 000544' 265 17 0 00 000200 BOD1 JSP 0,11D INPUT CHARACTER NO.
625 000545' 231 92 0 00 000120 BCD1 IDIVI B,*0401 IDIVIDE TO GET (BLOCK NO. - 1)
626 000546' 350 08 0 00 000002 AGS 8 ADD ONE TO GET BLOCK NO.
627 000547' 074 81 0 02 000000 USEQ 1, (B) INPUT 1.
628 000550' 066 31 0 00 000000 INPUT 1.
629 000551' 202 92 0 00 000025 MOVEM B,OSKBLK NEW BLOCK NO.
630 000552' 230 19 0 00 000002 MOVE 15,16UF *2 GET NEW BYTE COUNT
631 000553' 203 14 0 00 000014 MOVE 14,16UF *1 GET NEW BYTE POINTER
632 000554' 274 15 0 00 000030 SUB 15,C SUBTRACT TO GET BYTE COUNT
633 000555' 231 03 0 00 000025 IDIVI C,5 IDIVIDE TO GET WORD COUNT
634 000556' 270 14 0 00 000003 ADD 14,C ADD WORD COUNT TO BYTE POINTER
635 000557' 322 04 0 00 000036 JUMPE D,FSTLIN GO TO FSTLIN IF AT BEGINNING OF FRAME
636 000558' 133 06 0 00 000014 IBP 14 INCREMENT BYTE POINTER
637 000559' 364 04 0 00 000001 SJSA D,-2 SUBTRACT ONE FROM REMAINING BYTE COUNT

SECTION L. INTERPRETS AND EXECUTES PAGE (PB) STATEMENTS.

640 000562' 265 17 0 00 000000 UCPMD1 JSP 0,PAG CALL THE PAGING ROUTINE
641 000563' 254 00 0 00 000044 JRST UNGAC1 GO TO UNGAC1 IN INTERP,MAC
642 000564' 265 17 0 00 000062 PCHM1 JSP 0,PAG CALL THE PAGING ROUTINE
643 000565' 254 00 0 00 000446 JRST GETACT GO TO GETACT IN INTERP,MAC

NO ERRORS DETECTED

PROGRAM BREAK IS 00264B

3K CORE USED
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TITLE CPROS</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>COMMENT %</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>WRITTEN BY GARRET A. VANDER LUGT COMPLETED 14-MAY-74</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PART OF LEARN, SHR, VERSION 3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CPROS,MAC PROCESSES CALC MODE STATEMENTS. THE PROCESSED</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>STATEMENT IS PLACED IN THE CALC STATEMENT BUFFER SO THAT IT</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CAN BE EXECUTED BY CALC,MAC. MUCH OF THE CODE IN CPROS,MAC</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>IS SIMILAR TO THE CODE IN CFRAM,MAC.</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ENTRY POINT TO CPROS,MAC</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>CPROS = FROM CALC,MAC (SECTION A), LOCATED IN SECTION A</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>EXTERNAL SUBROUTINES CALLED BY CPROS,MAC</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>EXPCOR = EXPANDS THE LOW SEGMENT IF NECESSARY. LOCATED IN</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>AUX1,MAC (SECTION M)</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>FLOUT = &quot;OUTPUTS&quot; THE FLOATING POINT NO. IN AC-A INTO THE</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>CALC STATEMENT BUFFER</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>IIC = INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B,</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>LOCATED IN AUX1,MAC (SECTION E)</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>IIC2 = INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B,</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>LOCATED IN AUX1,MAC (SECTION E)</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>IODE = &quot;OUTPUTS&quot; THE INTEGER IN AC-A INTO THE CALC STATEMENT</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>BUFFER, LOCATED IN AUX1,MAC (SECTION D)</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>TTYIN = INPUT A FLOATING POINT NO. FROM A USER'S TERMINAL INTO</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>AC-B, LOCATED IN AUX1,MAC (SECTION F)</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOW1,MAC)</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>ADDR = ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>VARIABLES USED IN STUDENT CALCULATION MODE</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>BUFPTN = BYTE POINTER TO CALC STATEMENT BUFFER</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>CLOSTR = CALC STATEMENT BUFFER (16 WORDS)</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>CONST*13 (#OSTACK) = OPERATOR STACK (34 WORDS)</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>CONST*59 (#HOLD) = USED TO HOLD OPERATOR PRECEDING IN ARRAY</td>
<td></td>
</tr>
</tbody>
</table>
VARIABLE
54
55
56
DATA = STARTING ADDRESS FOR STORAGE OF STUDENT VARIABLES,
57
LESSON VARIABLES, AND STUDENT PERFORMANCE DATA
58
59
60
IDX1 = ADDRESS OF 1ST INDEX VARIABLE
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
IDX2 = ADDRESS OF 2ND INDEX VARIABLE
IDX2V = INITIAL VALUE OF 2ND INDEX AND FINAL VALUE OF 2ND INDEX
PC4 = SUBROUTINES STORE RETURN ADDRESS HERE
PC5 = SUBROUTINES STORE RETURN ADDRESS HERE
POLLIST (=SSTACK) = STRING STACK (72 WORDS)
PSTACK = POLISH STACK (72 WORDS)
PTYPE = POLISH TYPE STACK (72 WORDS)
RSVNUM = TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE
STORAGE VARIABLE
STYPE = STRING TYPE STACK (72 WORDS)
SYMDX = SYMBOL TABLE INDEX
SYMTAB = SYMBOL TABLE (17 WORDS)
DESCRIPTION OF CFLAGS STORED IN AC-16
BIT 17 COMMAND LEVEL
BIT 27 REPLACEMENT
BIT 28 ARRAY INDEX
BIT 29 RELATIONAL SYMBOL
BIT 33 2ND INDEX
BIT 34 INSIDE ITERATION COMPLETE
BIT 35 OUTSIDE ITERATION COMPLETE

!WISEG

ENTRY CPROS

EXTERN I1C, I1C2, FLOUT, IODC, TTYIN
EXTERN SYMDX, STYPE, PSTACK, PTYPE, POLLIST, CONST
EXTERN PC4, PC5, DATA, IDX1, IDX2, IDX2V, EXPGOR
EXTERN CLOST, ADDR, RSVNUM, SYMTAB, BUFNPT
SECTION D. EXECUTES ARRAY STATEMENTS.

ARRAY1 TCALL 4.A IINPUT CHARACTER

CALL A.11
CALL A.48
JMP A.11

ARRAY2 TCALL 4.A IINPUT CHARACTER

CALL A.11
CALL A.48
JMP A.11
I

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION G. PROCESSES ATYPE STATEMENTS.

671  007755  251 3 0 0 0 003055  ATYPE1 MOVE 4,5 INPUT ATYPE STATEMENT CODE IN AC-A

672  007964  133 0 0 0 003752  IDPB A,BUFPCNT INPUT CODE 0,05

673  007767  051 4 0 0 0 00201  IDPB A,BUFPCNT INPUT CHARACTER

674  007962  301 1 0 0 0 003101  IDPB A,"A"

675  007967  301 0 0 0 0 003301  IDPB A,"B"

676  007968  301 0 0 0 0 003121  IDPB A,"C"

677  007974  301 0 0 0 0 003321  IDPB A,"D"

678  007975  301 0 0 0 0 003331  IDPB A,"E"

679  007976  301 0 0 0 0 003341  IDPB A,"F"

680  007977  301 0 0 0 0 003351  IDPB A,"G"

681  007978  301 0 0 0 0 003361  IDPB A,"H"

682  007979  301 0 0 0 0 003371  IDPB A,"I"

683  007980  301 0 0 0 0 003381  IDPB A,"J"

684  007981  301 0 0 0 0 003391  IDPB A,"K"

685  007982  301 0 0 0 0 003401  IDPB A,"L"

686  007983  301 0 0 0 0 003411  IDPB A,"M"

687  007984  301 0 0 0 0 003421  IDPB A,"N"

688  007985  301 0 0 0 0 003431  IDPB A,"O"

689  007986  301 0 0 0 0 003441  IDPB A,"P"

690  007987  301 0 0 0 0 003451  IDPB A,"Q"

691  007988  301 0 0 0 0 003461  IDPB A,"R"

692  007989  301 0 0 0 0 003471  IDPB A,"S"

693  007990  301 0 0 0 0 003481  IDPB A,"T"

694  007991  301 0 0 0 0 003491  IDPB A,"U"

695  007992  301 0 0 0 0 003501  IDPB A,"V"

696  007993  301 0 0 0 0 003511  IDPB A,"W"

697  007994  301 0 0 0 0 003521  IDPB A,"X"

698  007995  301 0 0 0 0 003531  IDPB A,"Y"

699  007996  301 0 0 0 0 003541  IDPB A,"Z"

700  007997  301 0 0 0 0 003551  IDPB A,"0"

701  007998  301 0 0 0 0 003561  IDPB A,"1"

702  007999  301 0 0 0 0 003571  IDPB A,"2"

703  008000  301 0 0 0 0 003581  IDPB A,"3"

704  008001  301 0 0 0 0 003591  IDPB A,"4"

705  008002  301 0 0 0 0 003601  IDPB A,"5"

706  008003  301 0 0 0 0 003611  IDPB A,"6"

707  008004  301 0 0 0 0 003621  IDPB A,"7"

708  008005  301 0 0 0 0 003631  IDPB A,"8"

709  008006  301 0 0 0 0 003641  IDPB A,"9"

710  008007  301 0 0 0 0 003651  IDPB A,"."
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>743</td>
<td>0010721</td>
<td>MOVE Q,PC4</td>
</tr>
<tr>
<td>744</td>
<td>0010722</td>
<td>GET RETURN ADDRESS</td>
</tr>
<tr>
<td>745</td>
<td>0010723</td>
<td>JRST 1</td>
</tr>
<tr>
<td>746</td>
<td>0010724</td>
<td>RETURN</td>
</tr>
<tr>
<td>747</td>
<td>0010725</td>
<td>FORMAT 4</td>
</tr>
<tr>
<td>748</td>
<td>0010726</td>
<td>TCALL 4,A</td>
</tr>
<tr>
<td>749</td>
<td>0010727</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>750</td>
<td>0010728</td>
<td>CAIE A.11</td>
</tr>
<tr>
<td>751</td>
<td>0010729</td>
<td>JST 4</td>
</tr>
<tr>
<td>752</td>
<td>0010730</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>753</td>
<td>0010731</td>
<td>CAIE A.7</td>
</tr>
<tr>
<td>754</td>
<td>0010732</td>
<td>JST 4</td>
</tr>
<tr>
<td>755</td>
<td>0010733</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>756</td>
<td>0010734</td>
<td>JRST 4</td>
</tr>
<tr>
<td>757</td>
<td>0010735</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>758</td>
<td>0010736</td>
<td>JRST 4</td>
</tr>
<tr>
<td>759</td>
<td>0010737</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>760</td>
<td>0010738</td>
<td>JRST 4</td>
</tr>
<tr>
<td>761</td>
<td>0010739</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>762</td>
<td>0010740</td>
<td>JRST 4</td>
</tr>
<tr>
<td>763</td>
<td>0010741</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>764</td>
<td>0010742</td>
<td>JRST 4</td>
</tr>
<tr>
<td>765</td>
<td>0010743</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>766</td>
<td>0010744</td>
<td>JRST 4</td>
</tr>
<tr>
<td>767</td>
<td>0010745</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>768</td>
<td>0010746</td>
<td>JRST 4</td>
</tr>
<tr>
<td>769</td>
<td>0010747</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>770</td>
<td>0010748</td>
<td>JRST 4</td>
</tr>
<tr>
<td>771</td>
<td>0010749</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>772</td>
<td>0010750</td>
<td>JRST 4</td>
</tr>
<tr>
<td>773</td>
<td>0010751</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>774</td>
<td>0010752</td>
<td>JRST 4</td>
</tr>
<tr>
<td>775</td>
<td>0010753</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>776</td>
<td>0010754</td>
<td>JRST 4</td>
</tr>
<tr>
<td>777</td>
<td>0010755</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>778</td>
<td>0010756</td>
<td>JRST 4</td>
</tr>
<tr>
<td>779</td>
<td>0010757</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>780</td>
<td>0010758</td>
<td>JRST 4</td>
</tr>
<tr>
<td>781</td>
<td>0010759</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>782</td>
<td>0010760</td>
<td>JRST 4</td>
</tr>
<tr>
<td>783</td>
<td>0010761</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>784</td>
<td>0010762</td>
<td>JRST 4</td>
</tr>
<tr>
<td>785</td>
<td>0010763</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>786</td>
<td>0010764</td>
<td>JRST 4</td>
</tr>
<tr>
<td>787</td>
<td>0010765</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>788</td>
<td>0010766</td>
<td>JRST 4</td>
</tr>
<tr>
<td>789</td>
<td>0010767</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>790</td>
<td>0010768</td>
<td>JRST 4</td>
</tr>
<tr>
<td>791</td>
<td>0010769</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>792</td>
<td>0010770</td>
<td>JRST 4</td>
</tr>
<tr>
<td>793</td>
<td>0010771</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>794</td>
<td>0010772</td>
<td>JRST 4</td>
</tr>
<tr>
<td>795</td>
<td>0010773</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
</tbody>
</table>

**SECTION H. PROCESSES ASSIGNMENT STATEMENTS.**
<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>CONTENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>002200</td>
<td>CIN A, &quot;A&quot;</td>
</tr>
<tr>
<td>002201</td>
<td>251 01 00 0802477</td>
</tr>
<tr>
<td>002202</td>
<td>TCALL 11:6</td>
</tr>
<tr>
<td>002203</td>
<td>RETURN</td>
</tr>
<tr>
<td>002204</td>
<td>SUBI B, 0</td>
</tr>
<tr>
<td>002205</td>
<td>JADD 40 TO VARIABLE CODE</td>
</tr>
<tr>
<td>002206</td>
<td>MOVEL 8, STACK (G)</td>
</tr>
<tr>
<td>002207</td>
<td>INPUT VARIABLE CODE IN STRING STACK</td>
</tr>
<tr>
<td>002208</td>
<td>MOVEL B, 8</td>
</tr>
<tr>
<td>002209</td>
<td>INPUT &quot;V&quot; IN STRING TYPE STACK</td>
</tr>
<tr>
<td>00220A</td>
<td>TCALL 4:A</td>
</tr>
<tr>
<td>00220B</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>00220C</td>
<td>CIN A, 11</td>
</tr>
<tr>
<td>00220D</td>
<td>CAI A, 48</td>
</tr>
<tr>
<td>00220E</td>
<td>254 00 00 080131</td>
</tr>
<tr>
<td>00220F</td>
<td>JRST 4:3</td>
</tr>
<tr>
<td>002210</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>002211</td>
<td>AOA A, POL2</td>
</tr>
<tr>
<td>002212</td>
<td>GO TO POL2</td>
</tr>
<tr>
<td>002213</td>
<td>VARPO</td>
</tr>
<tr>
<td>002214</td>
<td>CIN A, &quot;A&quot;</td>
</tr>
<tr>
<td>002215</td>
<td>CAI A, 12</td>
</tr>
<tr>
<td>002216</td>
<td>JRST FERR</td>
</tr>
<tr>
<td>002217</td>
<td>IF CHARACTER IS NOT A LETTER, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>002218</td>
<td>MOVEL B, 8</td>
</tr>
<tr>
<td>002219</td>
<td>INPUT BYTE POINTER IN AC-B</td>
</tr>
<tr>
<td>00221A</td>
<td>SETS 0</td>
</tr>
<tr>
<td>00221B</td>
<td>IDPB A, 8</td>
</tr>
<tr>
<td>00221C</td>
<td>INPUT CHARACTER IN AC-F</td>
</tr>
<tr>
<td>00221D</td>
<td>TCALL 4:A</td>
</tr>
<tr>
<td>00221E</td>
<td>INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>00221F</td>
<td>CAI A, 13</td>
</tr>
<tr>
<td>002220</td>
<td>253 01 00 080132</td>
</tr>
<tr>
<td>002221</td>
<td>JRST 4:1</td>
</tr>
<tr>
<td>002222</td>
<td>IF CHARACTER IS NOT A NUMERAL, JUMP</td>
</tr>
<tr>
<td>002223</td>
<td>JRST AORF</td>
</tr>
<tr>
<td>002224</td>
<td>ELSE, GO TO AORF</td>
</tr>
<tr>
<td>002225</td>
<td>CIN A, &quot;B&quot;</td>
</tr>
<tr>
<td>002226</td>
<td>CAI A, 20</td>
</tr>
<tr>
<td>002227</td>
<td>JRST RVAR</td>
</tr>
<tr>
<td>002228</td>
<td>ELSE, GO TO RVAR</td>
</tr>
<tr>
<td>002229</td>
<td>JRST 8:1</td>
</tr>
<tr>
<td>00222A</td>
<td>IF CHARACTER IS NOT A LETTER, JUMP</td>
</tr>
<tr>
<td>00222B</td>
<td>JRST AORF</td>
</tr>
<tr>
<td>00222C</td>
<td>ELSE, GO TO AORF</td>
</tr>
<tr>
<td>00222D</td>
<td>CIN A, &quot;B&quot;</td>
</tr>
<tr>
<td>00222E</td>
<td>CAI A, 20</td>
</tr>
<tr>
<td>00222F</td>
<td>JRST RVAR</td>
</tr>
<tr>
<td>002230</td>
<td>ELSE, GO TO RVAR</td>
</tr>
<tr>
<td>002231</td>
<td>IF CHARACTER IS NOT A LETTER, GO TO RVAR</td>
</tr>
<tr>
<td>002232</td>
<td>JRST FUNC</td>
</tr>
<tr>
<td>002233</td>
<td>GO TO FUNC</td>
</tr>
<tr>
<td>002234</td>
<td>RVARI</td>
</tr>
<tr>
<td>002235</td>
<td>IDPB A, 8</td>
</tr>
<tr>
<td>002236</td>
<td>INPUT CHARACTER IN AC-F</td>
</tr>
<tr>
<td>002237</td>
<td>TCALL 4:A</td>
</tr>
<tr>
<td>002238</td>
<td>INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>002239</td>
<td>RVARI</td>
</tr>
<tr>
<td>00223A</td>
<td>MOVEL D, SYMIX (2)</td>
</tr>
<tr>
<td>00223B</td>
<td>GET INDEX TO END OF SYMBOL TABLE</td>
</tr>
<tr>
<td>00223C</td>
<td>CIN B, F</td>
</tr>
<tr>
<td>00223D</td>
<td>SYMIX (2) IS SYMIX FROM SYMBOL TABLE</td>
</tr>
<tr>
<td>00223E</td>
<td>CAI A, 50</td>
</tr>
<tr>
<td>00223F</td>
<td>JRST 3:3</td>
</tr>
<tr>
<td>002240</td>
<td>IF SYMIX NAME, JUMP</td>
</tr>
<tr>
<td>002241</td>
<td>SOUC 0, RV1</td>
</tr>
<tr>
<td>002242</td>
<td>GO CHECK NEXT SYMBOL, IF ANY</td>
</tr>
<tr>
<td>002243</td>
<td>JRST 1:3</td>
</tr>
<tr>
<td>002244</td>
<td>IF SYMBOL NOT FOUND, GO TO RV1</td>
</tr>
<tr>
<td>002245</td>
<td>HIRE 2, RV1</td>
</tr>
<tr>
<td>002246</td>
<td>IF SYMIX TABLE</td>
</tr>
<tr>
<td>002247</td>
<td>MOVEL C, STACK (G)</td>
</tr>
<tr>
<td>002248</td>
<td>PUT VALUE IN STRING STACK</td>
</tr>
<tr>
<td>002249</td>
<td>R1V1</td>
</tr>
<tr>
<td>00224A</td>
<td>MOVEL C, &quot;V&quot;</td>
</tr>
<tr>
<td>00224B</td>
<td>JRST 4:2</td>
</tr>
<tr>
<td>00224C</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>Address</td>
<td>Byte 1</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>1061</td>
<td>001544</td>
</tr>
<tr>
<td>1062</td>
<td>001545</td>
</tr>
<tr>
<td>1063</td>
<td>001546</td>
</tr>
<tr>
<td>1064</td>
<td>001547</td>
</tr>
<tr>
<td>1065</td>
<td>001550</td>
</tr>
<tr>
<td>1066</td>
<td>001551</td>
</tr>
<tr>
<td>1067</td>
<td>001552</td>
</tr>
<tr>
<td>1068</td>
<td>001553</td>
</tr>
<tr>
<td>1069</td>
<td>001554</td>
</tr>
<tr>
<td>1070</td>
<td>001555</td>
</tr>
<tr>
<td>1071</td>
<td>001556</td>
</tr>
<tr>
<td>1072</td>
<td>001557</td>
</tr>
<tr>
<td>1073</td>
<td>001558</td>
</tr>
<tr>
<td>1074</td>
<td>001559</td>
</tr>
<tr>
<td>1075</td>
<td>001560</td>
</tr>
<tr>
<td>1076</td>
<td>001561</td>
</tr>
<tr>
<td>1077</td>
<td>001562</td>
</tr>
<tr>
<td>1078</td>
<td>001563</td>
</tr>
<tr>
<td>1079</td>
<td>001564</td>
</tr>
<tr>
<td>1080</td>
<td>001565</td>
</tr>
<tr>
<td>1081</td>
<td>001566</td>
</tr>
<tr>
<td>1082</td>
<td>001567</td>
</tr>
<tr>
<td>1083</td>
<td>001568</td>
</tr>
<tr>
<td>1084</td>
<td>001569</td>
</tr>
<tr>
<td>1085</td>
<td>001570</td>
</tr>
<tr>
<td>1086</td>
<td>001571</td>
</tr>
<tr>
<td>1087</td>
<td>001572</td>
</tr>
<tr>
<td>1088</td>
<td>001573</td>
</tr>
<tr>
<td>1089</td>
<td>001574</td>
</tr>
<tr>
<td>1090</td>
<td>001575</td>
</tr>
<tr>
<td>1091</td>
<td>001576</td>
</tr>
<tr>
<td>1092</td>
<td>001577</td>
</tr>
<tr>
<td>1093</td>
<td>001578</td>
</tr>
<tr>
<td>1094</td>
<td>001579</td>
</tr>
<tr>
<td>1095</td>
<td>001580</td>
</tr>
<tr>
<td>1096</td>
<td>001581</td>
</tr>
<tr>
<td>1097</td>
<td>001582</td>
</tr>
<tr>
<td>1098</td>
<td>001583</td>
</tr>
<tr>
<td>1099</td>
<td>001584</td>
</tr>
<tr>
<td>1100</td>
<td>001585</td>
</tr>
<tr>
<td>1101</td>
<td>001586</td>
</tr>
<tr>
<td>1102</td>
<td>001587</td>
</tr>
<tr>
<td>1103</td>
<td>001588</td>
</tr>
<tr>
<td>1104</td>
<td>001589</td>
</tr>
<tr>
<td>1105</td>
<td>001590</td>
</tr>
<tr>
<td>1106</td>
<td>001591</td>
</tr>
<tr>
<td>1107</td>
<td>001592</td>
</tr>
<tr>
<td>1108</td>
<td>001593</td>
</tr>
<tr>
<td>1109</td>
<td>001594</td>
</tr>
<tr>
<td>1110</td>
<td>001595</td>
</tr>
<tr>
<td>1111</td>
<td>001596</td>
</tr>
<tr>
<td>1112</td>
<td>001597</td>
</tr>
<tr>
<td>1113</td>
<td>001598</td>
</tr>
</tbody>
</table>
SECTION J. PLACES VARIABLE IN SYMBOL TABLE AND OUTPUTS ITS ADDRESS.

ASVAR: MOVE K,A
MOVE D,SYMIOX
GET POINTER TO END OF SYMBOL TABLE

ASVII: HALT, A, SYMTAB (D) GET SYMBOL FROM SYMBOL TABLE

ASVIII: CANN A, 0
JRST, '0' IS VARIABLE NAME SAME AS SYMBOL?

ASVIII: JOST II YES, JUMP

INDEX: MOVE A,2
GET VARIABLE CODE <02> IN AC-A

IDPB A, B, BUFPT
OUTPUT <02>

MOVE A, P, STACK (F) GET VARIABLE ADDRESS FROM POLISH STACK

ADDI A, 128
ADDI B, 128

IDPB A, BUFPT
OUTPUT 1ST DIGIT OF ADDRESS (BASE 8)

MOVE A, 160
IDPB A,BUFPT
OUTPUT <20>

MOVH A, P, STACK (F) GET ABSOLUTE VALUE OF VARIABLE CODE
IDPB A, BUFPT
OUTPUT CODE FOR THIS VARIABLE

ADJA F, OUT1
GO TO OUT1

MOVE A, 21
OUTPUT VARIABLE CODE <21> IN AC-A

IDPB A, BUFPT
OUTPUT CODE

HRZ A, P, STACK (F) GET ADDRESS OF VARIABLE FROM POLISH STACK

ADDI A, 128
ADDI B, 128

IDPB A, BUFPT
OUTPUT 2ND DIGIT OF ADDRESS (BASE 8)

 Mojv A, 160
 IDPB A, BUFPT
 OUTPUT 2ND DIGIT OF COLUMN COUNT (BASE 8)

ADJA F, OUT1
GO TO OUT1

MOVE A, 160
OUTPUT 1ST DIGIT OF COLUMN COUNT (BASE 98)

IDPB A, BUFPT
OUTPUT <20>

MOVE A, 160
OUTPUT CHARACTOR THAT TERMINATED EXPRESSION IN AC-A

JST 0055 RETURN
<table>
<thead>
<tr>
<th>ADDR</th>
<th>126#</th>
<th>134#</th>
<th>135#</th>
<th>136#</th>
<th>138#</th>
<th>140#</th>
<th>141#</th>
<th>142#</th>
<th>143#</th>
<th>145#</th>
<th>153#</th>
<th>154#</th>
<th>157#</th>
<th>158#</th>
<th>159#</th>
</tr>
</thead>
<tbody>
<tr>
<td>130#</td>
<td>133#</td>
<td>134#</td>
<td>135#</td>
<td>136#</td>
<td>138#</td>
<td>140#</td>
<td>141#</td>
<td>142#</td>
<td>143#</td>
<td>145#</td>
<td>153#</td>
<td>154#</td>
<td>157#</td>
<td>158#</td>
<td>159#</td>
</tr>
<tr>
<td>140#</td>
<td>143#</td>
<td>144#</td>
<td>145#</td>
<td>146#</td>
<td>148#</td>
<td>149#</td>
<td>150#</td>
<td>151#</td>
<td>152#</td>
<td>154#</td>
<td>162#</td>
<td>163#</td>
<td>166#</td>
<td>167#</td>
<td>168#</td>
</tr>
<tr>
<td>150#</td>
<td>153#</td>
<td>154#</td>
<td>155#</td>
<td>156#</td>
<td>158#</td>
<td>159#</td>
<td>160#</td>
<td>161#</td>
<td>162#</td>
<td>164#</td>
<td>172#</td>
<td>173#</td>
<td>176#</td>
<td>177#</td>
<td>178#</td>
</tr>
<tr>
<td>260#</td>
<td>263#</td>
<td>264#</td>
<td>265#</td>
<td>266#</td>
<td>268#</td>
<td>269#</td>
<td>270#</td>
<td>271#</td>
<td>272#</td>
<td>274#</td>
<td>282#</td>
<td>283#</td>
<td>286#</td>
<td>287#</td>
<td>288#</td>
</tr>
<tr>
<td>356#</td>
<td>359#</td>
<td>360#</td>
<td>361#</td>
<td>362#</td>
<td>364#</td>
<td>365#</td>
<td>366#</td>
<td>367#</td>
<td>368#</td>
<td>370#</td>
<td>378#</td>
<td>379#</td>
<td>382#</td>
<td>383#</td>
<td>384#</td>
</tr>
<tr>
<td>436#</td>
<td>439#</td>
<td>440#</td>
<td>441#</td>
<td>442#</td>
<td>444#</td>
<td>445#</td>
<td>446#</td>
<td>447#</td>
<td>448#</td>
<td>449#</td>
<td>451#</td>
<td>452#</td>
<td>454#</td>
<td>455#</td>
<td>456#</td>
</tr>
<tr>
<td>463#</td>
<td>466#</td>
<td>467#</td>
<td>468#</td>
<td>469#</td>
<td>470#</td>
<td>471#</td>
<td>472#</td>
<td>473#</td>
<td>474#</td>
<td>475#</td>
<td>477#</td>
<td>478#</td>
<td>480#</td>
<td>481#</td>
<td>482#</td>
</tr>
<tr>
<td>546#</td>
<td>549#</td>
<td>550#</td>
<td>551#</td>
<td>552#</td>
<td>554#</td>
<td>555#</td>
<td>556#</td>
<td>557#</td>
<td>558#</td>
<td>559#</td>
<td>561#</td>
<td>562#</td>
<td>564#</td>
<td>565#</td>
<td>566#</td>
</tr>
<tr>
<td>629#</td>
<td>632#</td>
<td>633#</td>
<td>634#</td>
<td>635#</td>
<td>636#</td>
<td>637#</td>
<td>638#</td>
<td>639#</td>
<td>640#</td>
<td>642#</td>
<td>644#</td>
<td>645#</td>
<td>647#</td>
<td>648#</td>
<td>649#</td>
</tr>
<tr>
<td>682#</td>
<td>685#</td>
<td>686#</td>
<td>687#</td>
<td>688#</td>
<td>689#</td>
<td>690#</td>
<td>691#</td>
<td>692#</td>
<td>693#</td>
<td>695#</td>
<td>702#</td>
<td>703#</td>
<td>704#</td>
<td>705#</td>
<td>706#</td>
</tr>
<tr>
<td>786#</td>
<td>787#</td>
<td>788#</td>
<td>789#</td>
<td>790#</td>
<td>791#</td>
<td>792#</td>
<td>793#</td>
<td>794#</td>
<td>795#</td>
<td>796#</td>
<td>797#</td>
<td>798#</td>
<td>799#</td>
<td>800#</td>
<td>801#</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>OPTOR</th>
<th>1178</th>
<th>1196#</th>
<th>1184</th>
<th>1203</th>
<th>1206</th>
<th>1213</th>
<th>1216</th>
<th>1233</th>
<th>1244</th>
<th>1254</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSTACK</td>
<td>121#</td>
<td>1180</td>
<td>1184</td>
<td>1203</td>
<td>1206</td>
<td>1213</td>
<td>1216</td>
<td>1233</td>
<td>1244</td>
<td>1254</td>
</tr>
<tr>
<td>CUT1</td>
<td>133#</td>
<td>1311</td>
<td>1323</td>
<td>1335</td>
<td>1340</td>
<td>1357</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUT2</td>
<td>136#</td>
<td>1192</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAROP</td>
<td>1198</td>
<td>1234#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PC4</td>
<td>105#</td>
<td>132</td>
<td>242</td>
<td>284</td>
<td>318</td>
<td>322</td>
<td>356</td>
<td>404</td>
<td>551</td>
<td>559</td>
</tr>
<tr>
<td>PC5</td>
<td>105#</td>
<td>816</td>
<td>816</td>
<td>828</td>
<td>1359</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POINT1</td>
<td>124#</td>
<td>152</td>
<td>180</td>
<td>267</td>
<td>336</td>
<td>426</td>
<td>487</td>
<td>683</td>
<td>1367</td>
<td>1428</td>
</tr>
<tr>
<td>POINT3</td>
<td>125#</td>
<td>792</td>
<td>923</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POINT4</td>
<td>126#</td>
<td>264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POL1</td>
<td>861</td>
<td>823#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POL2</td>
<td>875</td>
<td>919</td>
<td>959</td>
<td>1088#</td>
<td>1108</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POL3</td>
<td>1114</td>
<td>1165#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLST</td>
<td>124#</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLY1</td>
<td>862</td>
<td>816#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLY2</td>
<td>864</td>
<td>816#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLY3</td>
<td>568</td>
<td>820#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRNERR</td>
<td>1237</td>
<td>1239#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSTACK</td>
<td>1044</td>
<td>1193</td>
<td>1218</td>
<td>1220</td>
<td>1224</td>
<td>1255</td>
<td>1316</td>
<td>1321</td>
<td>1324</td>
<td>1326</td>
</tr>
<tr>
<td>PTYPE</td>
<td>124#</td>
<td>1191</td>
<td>1226</td>
<td>1257</td>
<td>1307</td>
<td>315</td>
<td>350</td>
<td>365</td>
<td>441</td>
<td>499</td>
</tr>
<tr>
<td>Q</td>
<td>1111</td>
<td>132</td>
<td>173</td>
<td>188</td>
<td>215</td>
<td>308</td>
<td>315</td>
<td>350</td>
<td>365</td>
<td>441</td>
</tr>
<tr>
<td>563</td>
<td>568</td>
<td>609</td>
<td>613</td>
<td>616</td>
<td>629</td>
<td>646</td>
<td>649</td>
<td>690</td>
<td>709</td>
<td>718</td>
</tr>
<tr>
<td>744</td>
<td>752</td>
<td>756</td>
<td>759</td>
<td>766</td>
<td>802</td>
<td>809</td>
<td>810</td>
<td>816</td>
<td>818</td>
<td>829</td>
</tr>
<tr>
<td>1251</td>
<td>1252</td>
<td>1265</td>
<td>1269</td>
<td>1273</td>
<td>1277</td>
<td>1281</td>
<td>1283</td>
<td>1287</td>
<td>1291</td>
<td>1294</td>
</tr>
<tr>
<td>1232</td>
<td>1083</td>
<td>1416</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QUOTE</td>
<td>502</td>
<td>509</td>
<td>637</td>
<td>652#</td>
<td>655</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELERR</td>
<td>1171</td>
<td>1129#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELOP</td>
<td>1094</td>
<td>1096</td>
<td></td>
<td></td>
<td>1128#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RLP1</td>
<td>1124</td>
<td>1137#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RLP1A</td>
<td>1141</td>
<td>1147#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RLP2</td>
<td>1144</td>
<td>1153#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPAR</td>
<td>1081</td>
<td>1115#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPALTG</td>
<td>1158#</td>
<td>1162</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSVNUM</td>
<td>106#</td>
<td>314</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRAC</td>
<td>1202</td>
<td>1215#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV1</td>
<td>945#</td>
<td>946</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV2</td>
<td>951#</td>
<td>969</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV3</td>
<td>949#</td>
<td>960#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RVVAR</td>
<td>934</td>
<td>942#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RVAR</td>
<td>937</td>
<td>933</td>
<td>944#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCORE</td>
<td>294</td>
<td>302</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCRERR</td>
<td>303</td>
<td>319#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SELECT</td>
<td>275</td>
<td>277</td>
<td>285#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETVAR</td>
<td>273</td>
<td>219</td>
<td>1487#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPOCK</td>
<td>837</td>
<td>899#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPOCK</td>
<td>926</td>
<td>911#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPOUT</td>
<td>1325</td>
<td>1336#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPOVAR</td>
<td>639</td>
<td>860#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSTACK</td>
<td>1096#</td>
<td>1247</td>
<td>951</td>
<td>952</td>
<td>991</td>
<td>994</td>
<td>1082</td>
<td>1049</td>
<td>1093</td>
<td>1898</td>
</tr>
<tr>
<td>Page</td>
<td>Line</td>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>FABS = FROM CALC.MAC (SECTION O), LOCATED IN SECTION A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>FACOS = FROM CALC.MAC (SECTION O), LOCATED IN SECTION M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>FASIN = FROM CALC.MAC (SECTION O), LOCATED IN SECTION N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>FASIND = FROM CALC.MAC (SECTION O), LOCATED IN SECTION O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>FATAN = FROM CALC.MAC (SECTION O), LOCATED IN SECTION L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>FATAND = FROM CALC.MAC (SECTION O), LOCATED IN SECTION D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>FCOS = FROM CALC.MAC (SECTION O), LOCATED IN SECTION J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>FCOSD = FROM CALC.MAC (SECTION O), LOCATED IN SECTION H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>FEXP = FROM CALC.MAC (SECTION O), LOCATED IN SECTION E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>FINT = FROM INTERP.MAC (SECTION F) AND CALC.MAC (SECTIONS F &amp; O), LOCATED IN SECTION B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>FLOG = FROM CALC.MAC (SECTION O), LOCATED IN SECTION F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>FNLOG = FROM CALC.MAC (SECTION O), LOCATED IN SECTION G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>FRAN = FROM CALC.MAC (SECTION O), LOCATED IN SECTION C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>FSIN = FROM CALC.MAC (SECTION O), LOCATED IN SECTION R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>FSIND = FROM CALC.MAC (SECTION O), LOCATED IN SECTION I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>FSQRT = FROM CALC.MAC (SECTION O), LOCATED IN SECTION D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOH1.MAC)

FIELD = USED TO STORE INTERMEDIATE RESULTS OF CALCULATIONS
RANNUM = INTEGER RANDOM NO, USED BY RANDOM FUNCTION

DESCRIPTION OF CFLAGS STORED IN AC-16

BIT 31 NEGATIVE ARGUMENT

0000001 MISEG

ENTRY FATAN
ISECTION D. SQUARE ROOT SUBROUTINE:

170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212

FUNCS
MACRO 47(204)-9 00117 2-AUG-74 PAGE 1-3

160 000001 202 01 00 000000
161 000026 554 02 00 000001
162 000031 132 02 00 000016
163 000031 505 01 00 000000
164 000032 132 01 00 000174
165 000033 140 01 00 000002
166 000034 254 28 0 17 000000
167 168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212

FSORTI JUMPE A:SQ2
FSORTI JUMPE A:NEGNUM
MOVE B,A
ASHC B,-33
SUBI B,281
ROT B,-1
LSH B,-43
ASH C,-10
FSC C,177 (B)
MOVE C,A
FMP C,SS1 (B)
FAD C,SS2 (B)
MOVE B,A
FDV B,C
FAD C,B
FSC C,-1
FDV A,C
FADR A,C
FSC A, (E)
JRST (Q)
SQ2: MOVEI A,0
JRST (Q)
NEGNUM TDCALL 3,3ASCIA /
SQUARE ROOT OF NEGATIVE NUMBER

ISECTION E. EXPONENTIAL SUBROUTINE.

213 000073 200 02 00 000001
214 000074 317 02 00 000154
215 000075 254 06 0 000005
216 000076 317 02 0 000153

FEXP: MOVE B,A
CAMGE B,E7
JRST QUT2
CAMG B,E7
<table>
<thead>
<tr>
<th>Line</th>
<th>Machine Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>008771</td>
<td>JAST EXP1</td>
</tr>
<tr>
<td>0002</td>
<td>001200</td>
<td>TICALL 3,[CASC1] /</td>
</tr>
<tr>
<td>0003</td>
<td>001200</td>
<td>EXPONENTIAL ARGUMENT TOO LARGE</td>
</tr>
<tr>
<td>0004</td>
<td>001200</td>
<td>HRLO! A,37777</td>
</tr>
<tr>
<td>0005</td>
<td>001200</td>
<td>JAST (0)</td>
</tr>
<tr>
<td>0006</td>
<td>001200</td>
<td>OUT21</td>
</tr>
<tr>
<td>0007</td>
<td>001200</td>
<td>TICALL 3,[CASC1] /</td>
</tr>
<tr>
<td>0008</td>
<td>001200</td>
<td>EXPONENTIAL ARGUMENT TOO LARGE (NEGATIVE)</td>
</tr>
<tr>
<td>0009</td>
<td>001200</td>
<td>SETZ A,0</td>
</tr>
<tr>
<td>000A</td>
<td>001200</td>
<td>JAST (0)</td>
</tr>
<tr>
<td>000B</td>
<td>001200</td>
<td>EXP11</td>
</tr>
<tr>
<td>000C</td>
<td>001200</td>
<td>SETZH FIELD</td>
</tr>
<tr>
<td>000D</td>
<td>001200</td>
<td>MOVE E,D</td>
</tr>
<tr>
<td>000E</td>
<td>001200</td>
<td>MUL A,B</td>
</tr>
<tr>
<td>000F</td>
<td>001200</td>
<td>TSC B,C</td>
</tr>
<tr>
<td>0010</td>
<td>001200</td>
<td>MUL C,E</td>
</tr>
<tr>
<td>0011</td>
<td>001200</td>
<td>ASHC CY=242 (B)</td>
</tr>
<tr>
<td>0012</td>
<td>001200</td>
<td>ASH H1</td>
</tr>
<tr>
<td>0013</td>
<td>001200</td>
<td>ASH D=18</td>
</tr>
<tr>
<td>0014</td>
<td>001200</td>
<td>TSC D,E</td>
</tr>
<tr>
<td>0015</td>
<td>001200</td>
<td>FAD D,E,F</td>
</tr>
<tr>
<td>0016</td>
<td>001200</td>
<td>FMP D,E</td>
</tr>
<tr>
<td>0017</td>
<td>001200</td>
<td>MOVE A,E2</td>
</tr>
<tr>
<td>0018</td>
<td>001200</td>
<td>FMP A,D</td>
</tr>
<tr>
<td>0019</td>
<td>001200</td>
<td>MOVE B,E3</td>
</tr>
<tr>
<td>001A</td>
<td>001200</td>
<td>FMP A,D</td>
</tr>
<tr>
<td>001B</td>
<td>001200</td>
<td>MOVE C,E</td>
</tr>
<tr>
<td>001C</td>
<td>001200</td>
<td>FAD A,E1</td>
</tr>
<tr>
<td>001D</td>
<td>001200</td>
<td>FDVH C,A</td>
</tr>
<tr>
<td>001E</td>
<td>001200</td>
<td>MOVE A,E6</td>
</tr>
<tr>
<td>001F</td>
<td>001200</td>
<td>FSC A,F</td>
</tr>
<tr>
<td>0020</td>
<td>001200</td>
<td>MOVE D,E</td>
</tr>
<tr>
<td>0021</td>
<td>001200</td>
<td>JAST (0)</td>
</tr>
</tbody>
</table>

...
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>319</td>
<td>600231</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>320</td>
<td>600232</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>321</td>
<td>600233</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>322</td>
<td>600234</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>323</td>
<td>600235</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>324</td>
<td>600236</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>325</td>
<td>600237</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>326</td>
<td>600238</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>327</td>
<td>600239</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>328</td>
<td>600240</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>329</td>
<td>600241</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>330</td>
<td>600242</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>331</td>
<td>600243</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>332</td>
<td>600244</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>333</td>
<td>600245</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>334</td>
<td>600246</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>335</td>
<td>600247</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>336</td>
<td>600248</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>337</td>
<td>600249</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>338</td>
<td>600250</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>339</td>
<td>600251</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>340</td>
<td>600252</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>341</td>
<td>600253</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>342</td>
<td>600254</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>343</td>
<td>600255</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>344</td>
<td>600256</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>345</td>
<td>600257</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>346</td>
<td>600258</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>347</td>
<td>600259</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>348</td>
<td>600260</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>349</td>
<td>600261</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>350</td>
<td>600262</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>351</td>
<td>600263</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>352</td>
<td>600264</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>353</td>
<td>600265</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>354</td>
<td>600266</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>355</td>
<td>600267</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>356</td>
<td>600268</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>357</td>
<td>600269</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>358</td>
<td>600270</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>359</td>
<td>600271</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>360</td>
<td>600272</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>361</td>
<td>600273</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>362</td>
<td>600274</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>363</td>
<td>600275</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>364</td>
<td>600276</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>365</td>
<td>600277</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>366</td>
<td>600278</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>367</td>
<td>600279</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>368</td>
<td>600280</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>369</td>
<td>600281</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>370</td>
<td>600282</td>
<td>MOVE B,A</td>
</tr>
<tr>
<td>371</td>
<td>600283</td>
<td>MOVE B,A</td>
</tr>
</tbody>
</table>
### SECTION L. ARCTAN (RADIANS) SUBROUTINE

<table>
<thead>
<tr>
<th>FUNC</th>
<th>MACRO 47(236)</th>
<th>200117</th>
<th>2-AUG-74</th>
<th>PAGE 1-7</th>
</tr>
</thead>
<tbody>
<tr>
<td>372</td>
<td>000272</td>
<td>607</td>
<td>0 0 0 003808</td>
<td>TLNN G,3900</td>
</tr>
<tr>
<td>373</td>
<td>000273</td>
<td>213</td>
<td>0 0 0 000002</td>
<td>MOVNS B</td>
</tr>
<tr>
<td>374</td>
<td>000274</td>
<td>325</td>
<td>0 0 0 000002</td>
<td>SKIP5 E</td>
</tr>
<tr>
<td>375</td>
<td>000275</td>
<td>607</td>
<td>0 0 0 003808</td>
<td>MOVNS B</td>
</tr>
<tr>
<td>376</td>
<td>000276</td>
<td>202</td>
<td>0 0 0 000002</td>
<td>MOYEM B,F</td>
</tr>
<tr>
<td>377</td>
<td>000277</td>
<td>164</td>
<td>0 0 0 000002</td>
<td>FMPI R B,B</td>
</tr>
<tr>
<td>378</td>
<td>000280</td>
<td>200</td>
<td>0 0 0 003326</td>
<td>MOVE A,SC9</td>
</tr>
<tr>
<td>379</td>
<td>000281</td>
<td>160</td>
<td>0 0 0 000002</td>
<td>FMPI R A,B</td>
</tr>
<tr>
<td>380</td>
<td>000302</td>
<td>140</td>
<td>0 0 0 003317</td>
<td>FAD A,SC7</td>
</tr>
<tr>
<td>381</td>
<td>000303</td>
<td>160</td>
<td>0 0 0 000002</td>
<td>FMPI R A,B</td>
</tr>
<tr>
<td>382</td>
<td>000304</td>
<td>140</td>
<td>0 0 0 003316</td>
<td>FAD A,SC5</td>
</tr>
<tr>
<td>383</td>
<td>000305</td>
<td>160</td>
<td>0 0 0 003315</td>
<td>FAD A,SC3</td>
</tr>
<tr>
<td>384</td>
<td>000306</td>
<td>140</td>
<td>0 0 0 000002</td>
<td>FMPI R A,B</td>
</tr>
<tr>
<td>385</td>
<td>000307</td>
<td>140</td>
<td>0 0 0 003325</td>
<td>FAD A,PIOT</td>
</tr>
<tr>
<td>386</td>
<td>000311</td>
<td>164</td>
<td>0 0 0 000004</td>
<td>S2B1</td>
</tr>
<tr>
<td>387</td>
<td>000312</td>
<td>254</td>
<td>0 0 0 000006</td>
<td>JRTA (T)</td>
</tr>
<tr>
<td>388</td>
<td>000313</td>
<td>200</td>
<td>0 0 0 000006</td>
<td>S3A1</td>
</tr>
<tr>
<td>389</td>
<td>000314</td>
<td>254</td>
<td>0 0 0 000006</td>
<td>JRTA (T)</td>
</tr>
<tr>
<td>390</td>
<td>000315</td>
<td>577265</td>
<td>210372</td>
<td>SC31</td>
</tr>
<tr>
<td>391</td>
<td>000316</td>
<td>175506</td>
<td>321276</td>
<td>SC51</td>
</tr>
<tr>
<td>392</td>
<td>000317</td>
<td>606315</td>
<td>946346</td>
<td>SC71</td>
</tr>
<tr>
<td>393</td>
<td>000325</td>
<td>164475</td>
<td>535792</td>
<td>SC91</td>
</tr>
<tr>
<td>394</td>
<td>000321</td>
<td>170039</td>
<td>000000</td>
<td>SP1</td>
</tr>
<tr>
<td>395</td>
<td>000322</td>
<td>200000</td>
<td>000000</td>
<td>SP31</td>
</tr>
<tr>
<td>396</td>
<td>000323</td>
<td>205753</td>
<td>000000</td>
<td>CD11</td>
</tr>
<tr>
<td>397</td>
<td>000324</td>
<td>205712</td>
<td>273466</td>
<td>SCD1</td>
</tr>
<tr>
<td>398</td>
<td>000329</td>
<td>201622</td>
<td>077829</td>
<td>PIOT1</td>
</tr>
<tr>
<td>399</td>
<td>000339</td>
<td>201622</td>
<td>077829</td>
<td>P10T1</td>
</tr>
</tbody>
</table>

**Notes:**
- Fatani, MOVAF B,A
- CAMC B,A1
- JRTA (T)
- CML E A
- CMAC B,A2
- JRTA (T)
- MOVSI C,281400
- CMAC B,C
- TREA E, -1
- FDVM C,B
- TLC E, (E)
- MOVE F,B
- FMP B,B
- MOV A,KB3
- FAD C,B
- MOVE A,KA3
- FDVM A,C
- FAD C,B
- MOVE A,KB2
- MOVE A,KA2
<table>
<thead>
<tr>
<th>FUNKS</th>
<th>MACRO 47(204)</th>
<th>20/11/74</th>
<th>2-AUG-74 PAGE 1-9</th>
</tr>
</thead>
<tbody>
<tr>
<td>478</td>
<td>002423</td>
<td>1 54 52 0 0 0 002001</td>
<td>FSBR B,A</td>
</tr>
<tr>
<td>479</td>
<td>002424</td>
<td>3 52 1 0 0 0 004407</td>
<td>JUMP B, ILLSIN</td>
</tr>
<tr>
<td>480</td>
<td>002425</td>
<td>5 52 1 0 0 0 004341</td>
<td>JUMPE B, RTANG</td>
</tr>
<tr>
<td>481</td>
<td>002426</td>
<td>7 52 1 0 0 0 004017</td>
<td>MOVE 12, A</td>
</tr>
<tr>
<td>482</td>
<td>002427</td>
<td>9 52 1 0 0 0 003022</td>
<td>MOVE A, B</td>
</tr>
<tr>
<td>483</td>
<td>002430</td>
<td>11 52 1 0 0 0 003035</td>
<td>JSP Q, FSORT</td>
</tr>
<tr>
<td>484</td>
<td>002431</td>
<td>13 52 1 0 0 0 002061</td>
<td>FDRM F, A</td>
</tr>
<tr>
<td>485</td>
<td>002432</td>
<td>15 52 1 0 0 0 003261</td>
<td>JSP Q, FATAN</td>
</tr>
<tr>
<td>486</td>
<td>002433</td>
<td>17 52 1 0 0 0 003010</td>
<td>JST (19)</td>
</tr>
<tr>
<td>487</td>
<td>002434</td>
<td>19 52 1 0 0 0 003295</td>
<td>RTANG MOVE A, PIOT</td>
</tr>
<tr>
<td>488</td>
<td>002435</td>
<td>21 52 1 0 0 0 002036</td>
<td>SKIPGE A</td>
</tr>
<tr>
<td>489</td>
<td>002436</td>
<td>23 52 1 0 0 0 000001</td>
<td>MOVNS A</td>
</tr>
<tr>
<td>490</td>
<td>002437</td>
<td>25 52 1 0 0 0 002000</td>
<td>JIST (1)</td>
</tr>
<tr>
<td>491</td>
<td>002440</td>
<td>27 52 1 0 0 0 005201</td>
<td>ILLSIN TACALL 3, TASC10</td>
</tr>
<tr>
<td>492</td>
<td>002441</td>
<td>29 52 1 0 0 0 000000</td>
<td>ILLEGAL ARCSIN ARGUMENT</td>
</tr>
<tr>
<td>493</td>
<td>002442</td>
<td>31 52 1 0 0 0 000000</td>
<td>JIST (1)</td>
</tr>
<tr>
<td>494</td>
<td>002443</td>
<td>33 52 1 0 0 0 000000</td>
<td>SETZ A</td>
</tr>
<tr>
<td>495</td>
<td>002444</td>
<td>35 52 1 0 0 0 000000</td>
<td>JST (1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iSECTION O</th>
<th>ARCTAN (DEGREES) SUBROUTINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td></td>
</tr>
<tr>
<td>501</td>
<td>002445</td>
</tr>
<tr>
<td>502</td>
<td>002446</td>
</tr>
<tr>
<td>503</td>
<td>002447</td>
</tr>
<tr>
<td>504</td>
<td>002448</td>
</tr>
<tr>
<td>505</td>
<td></td>
</tr>
<tr>
<td>506</td>
<td></td>
</tr>
<tr>
<td>507</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iSECTION P</th>
<th>ARCCOS (DEGREES) SUBROUTINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>508</td>
<td></td>
</tr>
<tr>
<td>509</td>
<td>002449</td>
</tr>
<tr>
<td>510</td>
<td>002450</td>
</tr>
<tr>
<td>511</td>
<td>002451</td>
</tr>
<tr>
<td>512</td>
<td></td>
</tr>
<tr>
<td>513</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td></td>
</tr>
<tr>
<td>515</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iSECTION Q</th>
<th>ARCSIN (DEGREES) SUBROUTINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>516</td>
<td></td>
</tr>
<tr>
<td>517</td>
<td>002452</td>
</tr>
<tr>
<td>518</td>
<td>002453</td>
</tr>
<tr>
<td>519</td>
<td>002454</td>
</tr>
<tr>
<td>520</td>
<td></td>
</tr>
<tr>
<td>521</td>
<td></td>
</tr>
<tr>
<td>522</td>
<td></td>
</tr>
<tr>
<td>523</td>
<td></td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 000531

3K CORE USED
TITLE AUX1

COMMENT %

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 15-MAY-74

PART OF LEARN.SHR, VERSION 3

AUX1.MAC CONTAINS A NUMBER OF SUBROUTINES WHICH ARE USED BY OTHER
SUBPROGRAMS IN THE LEARN.SHR HIGH SEGMENT.

CONTENTS OF AUX1.MAC

SECTION A, INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL
THE END OF THE LINE IS ENCOUNTERED.

SECTION B, FILLS THE DISK INPUT BUFFER FOR THE DISK INPUT
SUBROUTINE.

SECTION C, OUTPUTS A CHARACTER FROM AC-A TO THE DISK.

SECTION D, OUTPUTS AN INTEGER IN AC-A.

SECTION E, INPUTS AN INTEGER INTO AC-B.

SECTION F, FLOATING POINT INPUT SUBROUTINE.

SECTION G, FLOATING POINT OUTPUT SUBROUTINE.

SECTION H, SUBROUTINE FOR EXPANDING THE LOW SEGMENT.

ENTRY POINTS TO AUX1.MAC

EOLIN  =  FROM LEARN.MAC (SECTIONS F, G, N & O), LOCATED IN
SECTION A

EXPCOR  =  FROM INTERP.MAC (SECTIONS A, B, GG & HH), CPROS.MAC
(SECTION C), AND ACTION.MAC (SECTIONS D & F), LOCATED
IN SECTION H

FLOUT  =  FROM CPROS.MAC (SECTION J), LOCATED IN SECTION G

GETBUF  = FROM INTERP.MAC (SECTION A), LOCATED IN SECTION B

I1C  =  FROM CPROS.MAC (SECTIONS F & G), LOCATED IN SECTION E

I1C2  = FROM CPROS.MAC (SECTIONS A, C, D, E, F & G), LOCATED
IN SECTION E

I1D  = FROM INTERP.MAC (SECTIONS B, EE & GG) AND ACTION.MAC
SECTION A. INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL AN END OF LINE IS ENCOUNTERED.

TICALL 4: A; INPUT CHARACTER

SECTION B. FILLS DISK INPUT BUFFER FOR DISK INPUT ROUTINE WHICH IS LOCATED IN FAST MEMORY.

GETBUF1 AOS DSKBLK; INCREMENT DISK BLOCK NO.

USETI 1; DSKBLK; TELL THE MONITOR WHICH BLOCK TO GET NEXT

INPUT DISK BLOCK

IF NO ERROR, GO TO RSTACS

IF ERROR, GO TO DISK INPUT ERROR ON CHANNEL 1 = AUX1

HALT
I MACRO 47(204)-7 08120 2-AUG-74 PAGE 1-3

160 000014 284 10 0 0 000001 RSTACSI MOVE 14 IWB 1 IPUT BUFFER BYTE COUNT IN AC-14
161 000015 284 19 0 0 000002 MOVEM 15 IWB 2 IPUT BUFFER BYTE COUNT IN AC-19
162 000016 254 00 0 0 000011 JRST II IGO GET CHARACTER FROM BUFFER

SECTION C. OUTPUTS A CHARACTER FROM AC-A TO THE DISK
ION CHANNEL 2.

169 000017 375 00 0 0 000022 IUTC3R IUSBG OSBF 2 IDECORENT BUFFER CHARACTER COUNT
170 000020 284 00 0 0 000023 JRST OUTBF IF BUFFER FULL, GO EMPTY IT
171 000021 135 01 0 0 000004 IPBF A 0OSBF 1 IPUT CHARACTER IN BUFFER
172 000022 254 00 0 0 000000 JRST (G) RETURN
173 000023 057 02 0 0 000008 PUTBF 1 OUT 2. IEMPTY BUFFER
174 000024 254 00 0 0 000017 JRST DUTC3R IPUT NEXT CHARACTER IN BUFFER

175 DISK OUTPUT ERROR ON CHANNEL 2 = AUX1

176 000029 051 03 0 0 000041 /

177 000026 254 04 0 0 000000 HALT

SECTION D. OUTPUTS AN INTEGER FROM AC-A TO A USER'S TERMINAL
FOR THE CALC STATEMENT BUFFER.

185 000027 200 04 0 0 000015 IDCC1 MOVE D 0 DCSBF 1BUFFN1 IPUT OUTPUT INSTRUCTION IN AC-D
186 000030 254 00 0 0 000022 JRST 2

187 000031 284 00 0 0 000015 IDIME1 MOVE 0 IDTCALL 1 A 3 IPUT OUTPUT INSTRUCTION IN AC-D
188 000032 220 04 0 0 000000 MOVEM D 10 IPUT INSTRUCTION IN 10
189 000033 284 00 0 0 000000 MOVEM D PDLIST 1 ISET UP PUSH DOWN POINTER IN AC-D
190 000034 231 01 0 0 000012 IDIVI A 12 IGEB DIGIT IN AC-B
191 000035 484 02 0 0 000060 IORD B 86 IORD TO ASCII CODE
192 000036 226 04 0 0 000022 RUSH D 8 IPUT CHARACTER IN PUSH DOWN LIST
193 000037 136 01 0 0 000034 JUMP 0 A 10 I 3 IGD GET NEXT DIGIT, IF ANY
194 000038 262 04 0 0 000001 POP D 24 IGET CHARACTER FROM PUSH DOWN LIST
195 000039 256 00 0 0 000032 XOT 10 IOUTPUT THE CHARACTER
196 000040 240 04 0 0 000000 TLNE 0777777 IANY CHARACTERS LEFT?
197 000041 254 00 0 0 000040 JRST 3 IYES, GO GET NEXT CHARACTER
198 000042 254 00 0 0 000086 JRST (G) IELSE, RETURN

SECTION E. INPUTS AN INTEGER FROM A USER'S TERMINAL OR THE DISK INTO AC-B.

205 000045 284 03 0 0 000017 IIDI MOVE 0 Q ISAVE RETURN ADDRESS
206 000046 284 02 0 0 000017 MOVEM B ITCALL 4 A 1 IPUT INPUT INSTRUCTION IN AC-B
207 000047 254 00 0 0 000022 JRST 3
208 000048 290 03 0 0 000007 IIII MOVE 0 0 ISAVE RETURN ADDRESS
209 000049 290 03 0 0 000021 MOVEM B DLEIS 0 01 IPUT INPUT INSTRUCTION IN AC-B
210 000050 254 00 0 0 000010 MOVEM B 10 IPUT INSTRUCTION IN 10
211 000051 284 02 0 0 000000 SETSB
212 000052 254 00 0 0 000052 IIDB1 XOT 10 IINPUT CHARACTER
IF K110<INF4: FLTR B+B CONVERT TO FLOATING POINT

INF71 FAD R0, B  ADD THIS VALUE TO NO. IN AC-0

SETR B,C

XCT 10  INPUT CHARACTER

CAIN A,"="  IS CHARACTER A MINUS?

SETO C  YES, SET AC-C TO ONES

CALL A,#"="

CAIN A,"

JST INF5  IF CHARACTER IS "+" OR "-", GO TO INF5

JST +2  ELSE, JUMP

XCT 10  INPUT CHARACTER

CALL A,#"0"

CALL A,#"9"

JST ALLN3  IF CHARACTER IS NOT A DIGIT, GO TO ALLN3

I MUL I B,12  SHIFTING, IN AC-B ONE DIGIT LEFT

ADDI B,-80(A)  ADD DIGIT TO NO. ALREADY IN AC-B

JST INF5  GO GET NEXT CHARACTER

ALLN3: JUMPE B, DGN  IF EXPONENT IS ZERO, GO TO DGN

CALL B,#040  B'S EXPONENT LESS THAN 427

MOVEI B,#D39  INO, SET IT TO 39

JSUMI B,NEGEX  IF EXPONENT IS NEGATIVE, GO TO NEGEX

DIVI B,12  SEPARATE EXPONENT INTO TWO PARTS

FMPE D, TENTB+1 (B)  MULTIPLY TO GET VALUE OF NUMBER

MOVNS C  MAKE OTHER PART OF EXPONENT NEGATIVE

FMPE D, TENTAB+0 (C)  MULTIPLY AGAIN TO GET FINAL VALUE OF NUMBER

DIVI D,12  SEPARATE EXPONENT INTO TWO PARTS

DIVD D, TENTAB+1 (B)  IDIVIDE TO COMPUTE VALUE OF NUMBER

MOVNS C  MAKE OTHER PART OF EXPONENT NEGATIVE

DIVD D, TENTAB+1 (B)  IDIVIDE TO COMPUTE VALUE OF NUMBER

DIVI D,12  SEPARATE EXPONENT INTO TWO PARTS

IDIVD D, TENTAB+0 (C)  IDIVIDE TO COMPUTE VALUE OF NUMBER

MOVNS C  MAKE OTHER PART OF EXPONENT NEGATIVE

JRST DGN  GO TO DGN

JRST 0  RETURN

THE ROUTINE BELOW CONVERTS THE NO. IN AC-B TO FLOATING POINT.

IT IS NEEDED ONLY FOR THE K128 CPU.

IFE K11B< FLOAT I DIVI B,#00000 I DIVIDE NO. INTO TWO PARTS

SHI B,1 الل ORDER PART IS ZERO

TLC B, 254088  CONVERT HIGH ORDER PART TO FLOATING POINT

TLC C, 233080  CONVERT LOW ORDER PART TO FLOATING POINT

FAD B,C  ADD THE TWO PARTS AND NORMALIZE

JRST (0)  RETURN

SECTION 6. OUTPUTS A FLOATING POINT NUMBER FROM AC-A TO A USER'S TERMINAL OR TO THE CALC STATEMENT BUFFER.
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION II. EXPANDS THE LOW SEGMENT IF NEEDED TO STORE I STUDENT PERFORMANCE DATA.

EXPAND MOVEI B, DATA+1(INDEX) GET HIGHEST LOWSEG ADDRESS TO BE REFERENCED

CORE B REQUEST MORE CORE

I TYPE ERROR MESSAGE IF CORE CAPACITY EXCEEDED

CALL 3 I ASCIZ /

CORE_CAPACITY_EXCEEDED

PROGRAM BREAK IS 00642

3K CORE USED
```
<table>
<thead>
<tr>
<th>LOW1</th>
<th>MACRO 47(204)-7 09122 2-AUG-74 PAGE 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>LOW1</td>
</tr>
<tr>
<td>COMMENT</td>
<td>%</td>
</tr>
<tr>
<td>WROTE</td>
<td>BY GARRET A. VANDER LOGT COMPLETED 15-MAY-74</td>
</tr>
<tr>
<td>LOWSEG</td>
<td>FOR LEARN, SHR, VERSION 3</td>
</tr>
<tr>
<td>LOW1.MAC IS A DUMMY LOW SEGMENT WHICH IS USED TO LABEL MEMORY</td>
<td></td>
</tr>
<tr>
<td>LOCATIONS IN THE LOW SEGMENT SO THAT THEY CAN BE EASILY REFERENCED</td>
<td></td>
</tr>
<tr>
<td>BY PROGRAMS IN THE LEARN HIGH SEGMENT, SINCE NO INSTRUCTIONS OR</td>
<td></td>
</tr>
<tr>
<td>CONSTANTS ARE STORED IN THE LOW SEGMENT, NO BINARY CORE IMAGE OF IT</td>
<td></td>
</tr>
<tr>
<td>IS PRODUCED BY THE SSAVE MONITOR COMMAND.</td>
<td></td>
</tr>
<tr>
<td>ENTRY</td>
<td>ADDR</td>
</tr>
<tr>
<td>SYMIDX,</td>
<td>SYTPE, PSTACK, PTYPE, SYMTAB, PCS, RGTANS</td>
</tr>
<tr>
<td>SNANS,</td>
<td>IDXSAV, TOL, ARGCT, IIDX, TAGREG, TAGCNT, TAGREP, TAGPNT</td>
</tr>
<tr>
<td>IDX1,</td>
<td>IDX1V, IDX2, IDX2Y, CLCSTR, PDSST, Const, FIELD, RANNUM, TMBLK</td>
</tr>
<tr>
<td>REVIDX,</td>
<td>REVNUM, SAVPDS, RSVNUM, PC3, PC4, CBSAV, COMMN, CVPSDS, JOBNO,</td>
</tr>
<tr>
<td>SAVPNT,</td>
<td>LESH, TSTW, RPTCNT, HTACNT, SUBPDP</td>
</tr>
<tr>
<td>SUBP,</td>
<td>OUTF, DELIN, COUNT, SANS, NUMPNT, STATE</td>
</tr>
<tr>
<td>DATA,</td>
<td>RNUN, WNM, NNUM, TNUM, ANSTIM, RUTST, TIME, FRHTOT</td>
</tr>
<tr>
<td>TAG,</td>
<td>POLST, SANS, LANS, RFSNUM, SFRH, SLDNUM, LKS4V</td>
</tr>
<tr>
<td>ISUBF,</td>
<td>OBUF, INAM, PPN, ONAM, BLOCK, STORE, PC2, ARGNT,</td>
</tr>
<tr>
<td>INTEXT,</td>
<td>INLOC, BUF, OBF, OFLAGS, DEVICE, AC14, AC15, DBLK, ID</td>
</tr>
<tr>
<td>000137</td>
<td>LOV0ER=137</td>
</tr>
<tr>
<td>000001</td>
<td>LOC LOV0ER</td>
</tr>
<tr>
<td>0000001</td>
<td>KMD 39,1</td>
</tr>
<tr>
<td>0000001</td>
<td>RELOC 0</td>
</tr>
<tr>
<td>000001</td>
<td>STORE: BLOCK 1 IUSED BY LEARN, MAC, INTERP, MAC, CALC, MAC, ACTION, MAC, AND AUI1, MAC</td>
</tr>
<tr>
<td>000001</td>
<td>ARGNT: BLOCK 1 IUSED BY LEARN, MAC AND INTERP, MAC</td>
</tr>
<tr>
<td>0000007</td>
<td>JOBNO, I BLOCK 1 IUSED BY LEARN, MAC</td>
</tr>
<tr>
<td>000010</td>
<td>TRMBLK1 BLOCK 3 IUSED BY LEARN, MAC AND INTERP, MAC</td>
</tr>
<tr>
<td>000013</td>
<td>INTEXT1 BLOCK 4 IUSED BY LEARN, MAC</td>
</tr>
<tr>
<td>000020</td>
<td>INTLOC BLOCK 12 IUSED BY LEARN, MAC</td>
</tr>
<tr>
<td>000032</td>
<td>IBUF: BLOCK 3 IUSED BY LEARN, MAC, ACTION, MAC, AND AUI1, MAC</td>
</tr>
<tr>
<td>000035</td>
<td>OBUF: BLOCK 3 IUSED BY LEARN, MAC AND AUI1, MAC</td>
</tr>
<tr>
<td>000040</td>
<td>INAM: BLOCK 4 IUSED BY LEARN, MAC, INTERP, MAC, AND CALC, MAC</td>
</tr>
<tr>
<td>000044</td>
<td>ONAM: BLOCK 4 IUSED BY LEARN, MAC AND INTERP, MAC</td>
</tr>
<tr>
<td>000050</td>
<td>PRT1: BLOCK 1 IUSED BY LEARN, MAC AND INTERP, MAC</td>
</tr>
<tr>
<td>000051</td>
<td>INBUF: BLOCK 223 IUSED BY LEARN, MAC</td>
</tr>
<tr>
<td>000254</td>
<td>OUTF: BLOCK 203 IUSED BY LEARN, MAC</td>
</tr>
<tr>
<td>000051</td>
<td>ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN</td>
</tr>
<tr>
<td>000051</td>
<td>JLDLOW, MAC, RPTLOW, MAC, AND PRL30W, MAC,</td>
</tr>
<tr>
<td>000051</td>
<td>PC21 BLOCK 1 IUSED BY INTERP, MAC</td>
</tr>
<tr>
<td>000051</td>
<td>PC31 BLOCK 1 IUSED BY CALC, MAC</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>LDI1</th>
<th>MACRO 47(204) - 6 2-AUG-74 PAGE 1-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>001163'</td>
</tr>
<tr>
<td>108</td>
<td>001164'</td>
</tr>
<tr>
<td>109</td>
<td>001166'</td>
</tr>
<tr>
<td>110</td>
<td>001167'</td>
</tr>
<tr>
<td>111</td>
<td>001170'</td>
</tr>
<tr>
<td>112</td>
<td>001171'</td>
</tr>
<tr>
<td>113</td>
<td>001172'</td>
</tr>
<tr>
<td>114</td>
<td>001174'</td>
</tr>
<tr>
<td>115</td>
<td>001176'</td>
</tr>
<tr>
<td>116</td>
<td>001177'</td>
</tr>
<tr>
<td>117</td>
<td>001200'</td>
</tr>
<tr>
<td>118</td>
<td>001201'</td>
</tr>
<tr>
<td>119</td>
<td>001202'</td>
</tr>
<tr>
<td>120</td>
<td>001312'</td>
</tr>
<tr>
<td>121</td>
<td>001425'</td>
</tr>
<tr>
<td>122</td>
<td>001591'</td>
</tr>
<tr>
<td>123</td>
<td>001592'</td>
</tr>
<tr>
<td>124</td>
<td>001593'</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 001594

3K CORE USED
TITLE LEDIT

COMMENT %

WRITTEN BY GARRETT A. VANDER LUGT  COMPLETED 14-JUL-73

PART OF LEDIT.SHR, VERSION 3

******************************************************************************

SPECIAL NOTE!

LEDIT.MAC CONTAINS THE ONLY ENTRY POINT TO THE LEDIT HIGH SEGMENT.
THIS ENTRY POINT IS LOCATED IN SECTION B OF THIS PROGRAM.

THE LEDIT HIGH SEGMENT IS COMPOSED OF THE FOLLOWING SUBPROGRAMS:

1. LEDIT.MAC
2. PERL.MAC
3. LPRINT.MAC
4. WRTE.MAC
5. AUX2.MAC
6. LDLYON.MAC (DUMMY LOW SEGMENT)

THESE PROGRAMS SHOULD BE LOADED IN THE ORDER IN WHICH THEY ARE LISTED
ABOVE. AFTER LOADING, A REENTRANT, BINARY CORE IMAGE OF THE LEDIT
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SSsave MONITOR COMMAND.

THE LEDIT HIGH SEGMENT EXECUTES THE FOLLOWING LEARN SYSTEM COMMANDS:

1. BYE
2. COPY
3. EDIT
4. FILES
5. KJOB
6. LIST
7. MONITOR
8. RENAME
9. WRITE

******************************************************************************

LEDIT.MAC INTERPRETS LEARN SYSTEM COMMANDS AND CALLS THE PROPER SUB-
PROGRAMS TO EXECUTE THEM. IF A COMMAND CANNOT BE EXECUTED BY THE
LEDIT HIGH SEGMENT, THEN LEDIT.MAC PLACES A COMMAND INDEX IN LOCATION
STORE AND REPLACES ITSELF WITH THE HIGH SEGMENT WHICH CAN EXECUTE THE
COMMAND. WHEN LEDIT.MAC IS ENTERED FROM ANOTHER HIGH SEGMENT, IT
RETRIEVES A COMMAND INDEX FROM LOCATION STORE AND CALLS THE PROPER
SUBPROGRAM TO EXECUTE THE COMMAND INDICATED BY THE COMMAND INDEX.
CONTENTS OF LEDIT.MAC

SECTION A. PERFORMS THE GETSEG UUD.

SECTION B. CONTAINS THE ENTRY POINT TO THE LEDIT HIGH SEGMENT.

SECTION C. INPUTS THE FIRST WORD IN A LEARN COMMAND.

SECTION D. INTERPRETS THE FIRST WORD IN A LEARN COMMAND.

SECTION E. EXECUTES A "MONITOR" COMMAND.

SECTION F. SETS UP GETSEG ARGUMENTS FOR THE LEARN HIGH SEGMENT.

SECTION G. SETS UP GETSEG ARGUMENTS FOR THE LRPRF HIGH SEGMENT.

SECTION H. SETS UP GETSEG ARGUMENTS FOR THE LEPRF HIGH SEGMENT.

SECTION I. CALLS THE PROPER LEDIT SUBPROGRAM TO EXECUTE A
LEARN COMMAND.

SECTION J. EXECUTES A "BYE" COMMAND.

SECTION K. EXECUTES A "KJOB" COMMAND.

ENTRY POINTS TO LEDIT.MAC

GETSG = FROM WRTED.MAC (SECTION K) IN RESPONSE TO AN
"ET" OR "EP" EDIT COMMAND, LOCATED IN SECTION A.

RETURN = FROM ANOTHER HIGH SEGMENT, LOCATED IN SECTION B.

EXTERNAL SUBROUTINES CALLED BY LEDIT.MAC

CPY = EXECUTES A "COPY" COMMAND, LOCATED IN PERL.MAC (SECTION D).

EDT = EXECUTES A "EDIT" COMMAND, LOCATED IN WRTED.MAC (SECTION A).

EDT2 = EXECUTES A TEST MODE IE COMMAND, LOCATED IN WRTED.MAC
(SECTION A).

EOLIN = INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE
END OF A LINE IS REACHED, LOCATED IN AUX2.MAC (SECTION A).

FILS = EXECUTES A "FILES" COMMAND, LOCATED IN PERL.MAC (SECTION A).

LST = EXECUTES A "LIST" COMMAND, LOCATED IN PERL.MAC (SECTION C).

RENAME = EXECUTES A "RENAME" COMMAND, LOCATED IN PERL.MAC (SECTION B).
WRT – EXECUTES A "WRITE" COMMAND, LOCATED IN WRTED, MAC (SECTION A)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LDTLOW, MAC)

ARGMNT – ARGUMENTS FOR GETSEG UDO (6 WORDS)

STORE = COMMAND INDEX IS SAVED HERE DURING A GETSEG

TRMLK = TRMP, ARGUMENTS (3 WORDS)

MISEG

EXTERN ARGMNT, CPY, EDIT, CDT, EOLN, FLS, LST, RENAM, STORE, WRT, TRMLK

SECTION A, GET THE PROPER HIGH SEGMENT TO EXECUTE THE COMMAND

WHICH WAS TYPED BY THE USER. EXECUTION RESUMES IN LEAR, MAC.

I LRPTE, MAC, OR LPROS, MAC.

NOTE! NO DATA OR INSTRUCTIONS SHOULD BE PLACED AHEAD OF

"GETSG", SINCE THE RETURN ADDRESS OF THE GETSEG UDO MUST BE

THE SAME IN ALL HIGH SEGMENTS.

SECTION B. RETRIEVE THE COMMAND INDEX WHICH WAS PLACED IN LOCATION

"STORE" BY THE PREVIOUS HIGH SEGMENT AND JUMP TO THE PROPER ROUTINE

TO EXECUTE THE COMMAND.

RETURN! MOVE B, STORE IGET THE COMMAND INDEX IN AC-B
LIST OF LEARN COMMANDS IN SIXBIT CODE

COMMANDS

SETE C

MOVSI 0/21

CMDIN2

MOVSI 0/21

CMDIN2

GET A LEARN COMMAND IN AC-G

DELETE CHARACTERS THAT NEED NOT BE TESTED

DOES USER'S COMMAND MATCH THE LEARN COMMAND?

NOUJ 4+4

JUMP FOUR LOCATIONS

IF THIS IS A 2ND MATCH, GO TYPE ERROR MESSAGE

SET AC-G TO ONCE TO INDICATE A MATCH

SAVE INDEX IN AC-F

INCREMEN INDEX AND GO TEST NEXT COMMAND

IF NO MATCH, GO TYPE ERROR MESSAGE (SECTION E)

JUMP TO PROPER ROUTINE TO EXECUTE THE COMMAND
1. JUMP TO THE PROPER ROUTINE TO EXECUTE THE COMMAND WHICH WAS TYPED BY THE USER.

2. NAME: JSP G, RENAME
   JGTO TO RENAME LESSON

3. FILE: JSP G, FILES
   JGTO TO LIST FILES

4. LIST: JSP G, LIST
   JGTO TO LIST LESSON

5. EDIT: JSP G, EDIT
   JGTO TO EDIT LESSON

6. WRITE: JSP G, WRITE
   JGTO TO WRITE LESSON

7. COPY: JSP G, COPY
   JGTO TO COPY FILE

8. EXECUTE THE BYE COMMAND BY RUNNING THE LOGOUT PROGRAM.

9. RUNARG: SIXBIT/SYS,

10. EIGHTBIT/LOGOUT

11. MOVE: A, RUNARG, INPUT ADDRESS OF RUN ARGUMENTS IN AC-A.

12. RUN A, IRUN THE LOGOUT SYSTEM PROGRAM

13. HALT

14. EXECUTE THE KJOB COMMAND BY RUNNING THE KJOB PROGRAM.

15. ARG1: SIXBIT/SYS,

16. SIXBIT/KJOB

17. KJOB1, JRST +2

18. TTCALL 4:A, INPUT NEXT CHARACTER IN COMMAND STRING

19. CAI.N A, 'A', 40, IS CHARACTER A SPACE?

20. JGTO Ai, 'A', 'YES, GO INPUT NEXT CHARACTER

21. CAI.N A, 'A', 42, 42, 'YES, GO TO KJ

22. KJ: INPUT LINE FEED

23. TTCALL 4:A, INPUT TYPE ERROR MESSAGE (SECTION 0)
TITLE PERL

COMMENT %

WRITTEN BY GARRETT A. VANDER LUGT  COMPLETED 30-JUL-73

PART OF LEDIT,SHR, VERSION 3

PERL.MAC EXECUTES "COPY", "FILES", "LIST", AND "RENAME" COMMANDS.

CONTENTS OF PERL.MAC

SECTION A, EXECUTES A "FILES" COMMAND,

SECTION B, EXECUTES A "RENAME" COMMAND.

SECTION C, EXECUTES A "LIST" COMMAND.

SECTION D, EXECUTES A "COPY" COMMAND.

ENTRY POINTS TO PERL.MAC

CPY - FROM LEDIT.MAC (SECTION 1), LOCATED IN SECTION D

FILS - FROM LEDIT.MAC (SECTION 1), LOCATED IN SECTION A

LST - FROM LEDIT.MAC (SECTION 1), LOCATED IN SECTION C

RENA - FROM LEDIT.MAC (SECTION 1), LOCATED IN SECTION B

EXTERNAL SUBROUTINES CALLED BY PERL.MAC

INT - INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B,
LOCATED IN AUX2.MAC (SECTION A)

INCHR - INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN
AUX2.MAC (SECTION B)

IOD - OUTPUTS AN INTEGER FROM AC-A TO THE DISK, LOCATED IN
AUX2.MAC (SECTION E)

IOD8 - OUTPUTS AN OCTAL NUMBER FROM AC-A TO THE DISK, LOCATED IN
AUX2.MAC (SECTION F)

IOT - OUTPUTS AN INTEGER FROM AC-A TO A USER'S TERMINAL, LOCATED
IN AUX2.MAC (SECTION G)

LPRINT - PLACES A LESSON LISTING FILE IN PRINT QUEUE, LOCATED IN
LPRINT.MAC (SECTION A)
OUTCHR - OUTPUTS A CHARACTER FROM AC=A TO THE DISK, LOCATED IN AUX2,MAG (SECTION D)

OUTSTR - OUTPUTS A STRING OF CHARACTERS IMMEDIATELY FOLLOWING THE CALLING INSTRUCTION TO THE DISK, LOCATED IN AUX2,MAG (SECTION D)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOTLOW,MAG)

EDTNUM - I/O COMMAND LIST (2 WORDS)
INAM = LOOKUP ARGUMENTS FOR DISK INPUT (4 WORDS)
INBUF = DISK BUFFER FOR DUMP MODE I/O (131 WORDS)
ONAM - ENTER ARGUMENTS FOR DISK OUTPUT (4 WORDS)
PCI = RETURN ADDRESS TO LEDIT,MAG
PPN - USER'S PROJ-PROG NUMBERS
STORE = NO. OF COPIES OF LESSON LISTING WHICH WILL BE PRINTED ON THE LINE PRINTER

DESCRIPTION OF FLAGS STORED IN AC=6

BIT 31 LEGAL SWITCH
BIT 32 RECORD FILE
BIT 33 ANSWER FILE
BIT 34 PROCESSED FILE
BIT 35 UNPROCESSED FILE

00001 MISEG

ENTRY CIPY,FLIS,LST,RENM

EXTERN EDTNUM,IT,INAM,INBUF,INCHR,IOD,IOGB,INT,LPRT,ONAM
EXTERN OUTCHR,OUTSTR,PCI,PPN,STORE

000800 FLAGS=0.

A=1
B=2
C=3
D=4
E=5
F=6
G=7
H=10
<table>
<thead>
<tr>
<th>PERL</th>
<th>MACRO</th>
<th>47(204) 7 20127 2-AUG-74 PAGE 1-7</th>
</tr>
</thead>
<tbody>
<tr>
<td>371</td>
<td>02324</td>
<td>254 0 0 0 0 0 01320</td>
</tr>
<tr>
<td>372</td>
<td>02324</td>
<td>254 0 0 0 0 0 00367</td>
</tr>
<tr>
<td>373</td>
<td>02324</td>
<td>254 0 0 0 0 0 00107</td>
</tr>
<tr>
<td>374</td>
<td>02324</td>
<td>254 0 0 0 0 0 00384</td>
</tr>
<tr>
<td>375</td>
<td>02324</td>
<td>254 0 0 0 0 0 00041</td>
</tr>
<tr>
<td>376</td>
<td>02324</td>
<td>254 0 0 0 0 0 00345</td>
</tr>
<tr>
<td>377</td>
<td>02324</td>
<td>254 0 0 0 0 0 00165</td>
</tr>
<tr>
<td>378</td>
<td>02324</td>
<td>254 0 0 0 0 0 00366</td>
</tr>
<tr>
<td>379</td>
<td>02324</td>
<td>254 0 0 0 0 0 00220</td>
</tr>
<tr>
<td>380</td>
<td>02324</td>
<td>254 0 0 0 0 0 00132</td>
</tr>
<tr>
<td>381</td>
<td>02324</td>
<td>254 0 0 0 0 0 00120</td>
</tr>
<tr>
<td>382</td>
<td>02324</td>
<td>254 0 0 0 0 0 00002</td>
</tr>
<tr>
<td>383</td>
<td>02324</td>
<td>254 0 0 0 0 0 00101</td>
</tr>
<tr>
<td>384</td>
<td>02324</td>
<td>254 0 0 0 0 0 00224</td>
</tr>
<tr>
<td>385</td>
<td>02324</td>
<td>254 0 0 0 0 0 00122</td>
</tr>
<tr>
<td>386</td>
<td>02324</td>
<td>254 0 0 0 0 0 00333</td>
</tr>
<tr>
<td>387</td>
<td>02324</td>
<td>254 0 0 0 0 0 00222</td>
</tr>
<tr>
<td>388</td>
<td>02324</td>
<td>254 0 0 0 0 0 00345</td>
</tr>
<tr>
<td>389</td>
<td>02324</td>
<td>254 0 0 0 0 0 00310</td>
</tr>
<tr>
<td>390</td>
<td>02324</td>
<td>254 0 0 0 0 0 00145</td>
</tr>
<tr>
<td>391</td>
<td>02324</td>
<td>201 0 0 0 0 0 00001</td>
</tr>
<tr>
<td>392</td>
<td>02324</td>
<td>100 0 0 0 0 0 0331</td>
</tr>
<tr>
<td>393</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>394</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>395</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>396</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>397</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>398</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>399</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>400</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>401</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>402</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>403</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>404</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>405</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>406</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>407</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>408</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>409</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>410</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>411</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>412</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>413</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>414</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>415</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>416</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>417</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>418</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>419</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>420</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>421</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>422</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
<tr>
<td>423</td>
<td>02324</td>
<td>100 0 0 0 0 0 0324</td>
</tr>
</tbody>
</table>

Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
<th>Column 6</th>
<th>Column 7</th>
<th>Column 8</th>
<th>Column 9</th>
<th>Column 10</th>
<th>Column 11</th>
<th>Column 12</th>
<th>Column 13</th>
<th>Column 14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column 1</td>
<td>Column 2</td>
<td>Column 3</td>
<td>Column 4</td>
<td>Column 5</td>
<td>Column 6</td>
<td>Column 7</td>
<td>Column 8</td>
<td>Column 9</td>
<td>Column 10</td>
<td>Column 11</td>
<td>Column 12</td>
<td>Column 13</td>
<td>Column 14</td>
</tr>
</tbody>
</table>
TITLE LPRINT

COMMENT X

PREPARED BY GARRETT A. VANDER LUGT   COMPLETED 31-JUL-73

PART OF LEDIT,SHR VERSION 3

SPECIAL NOTE: THIS SUBROUTINE IS A MODIFIED VERSION OF A SUBROUTINE WHICH WAS WRITTEN BY NORMAN D. GRANT AT THE COMPUTER CENTER, WESTERN MICHIGAN UNIVERSITY.

LPRINT.MAC PLACES LESSON LISTINGS IN PRINT QUEUE SO THAT THEY WILL BE PRINTED ON THE LINE PRINTER.

CONTENTS OF LPRINT.MAC

SECTION A. SET UP QUEUING PARAMETERS,

SECTION B. OUTPUT QUEUING PARAMETERS TO THE DISK.

SECTION C. WAKE UP THE SPOOLING PROGRAM.

ENTRY POINT TO LPRINT.MAC

LPRINT - FROM PERL.MAC (SECTION C), LOCATED IN SECTION A

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOTLOW.MAC)

LINK (#THSJOB) = USER'S JOB NUMBER

LINK+1 (#QUEDIR) = NAME OF QUEUING UFO

LINK+2 (#OPNBLK) = ARGUMENTS FOR OPEN UFO (3 WORDS)

LINK+5 (#FILNAM) = FILENAME OF PARAMETER FILE

LINK+11 (#OLIST) = OUTPUT COMMAND LIST (2 WORDS)

LINK+13 (#OLESPEC) = EXTENDED LOOKUP ARGUMENTS (14 WORDS)

OAH = CONTAINS NAME OF FILE TO BE QUEUED

SPACE (#PARAM) = LIST OF QUEUING PARAMETERS (32 WORDS)

STORE = CONTAINS NUMBER OF COPIES TO BE PRINTED

%
LPRINT MACRO 47(284)-F 96129 2-AUG-74 PAGE 1-1

I; Entry LPRINT
EXTERN LINK,ONAM,SPACE,STORE

THJSJOB=LINK+1
QEDIR=LINK+1
OPNBK=LINK+2
FLNAH=LINK+3
GLIST=LINK+11
GLSPEC=LINK+13
PARAM=SPACE

A#1
B#2
C#3
D#4
E#5
BP#6
BP#7
CT#10
P#15
Q#17

SECTION A. SET UP QUEUING PARAMETERS IN CORE.
OPN1 SIXBIT / DSK/

LPRINT GEETAB A, 434, 113
GET MONITOR VERSION NO.

HALT

HRHES A

CAGE A, 50500

IS MONITOR VERSION 5.03 OR LATER?

HALT

INQ, HALT

OPEN B, OPN

MOVE I A, 40

CLEAR QUEUING PARAMETERS

SETAM PARA=-1(A)

MOVEI A, 16

MOVEN A, GLSPEC

MOVE 14 TO GLSPEC FOR EXTENDED LOOKUP

MOVE 1 TO GLSPEC IN USER'S OWN UFU

MOVE A, ONAM

GET FILENAME OF LIST FILE

MOVE A, GLSPEC+1

MOVE IT TO LOOKUP EFFECTIVE ADDRESS

MOVE A, PARAH+3

MOVE IT TO PARAMETER LIST ALSO

MOVE A, GLSPEC+3

MOVE EXT. TO LOOKUP EFFECTIVE_ADDRESS

MOVE A, PARA=34

MOVE EXT. TO PARAMETER LIST

OPEN B, OPN

OPEN CHANNEL B, DUMP MODE
SECTION B. WRITE PARAMETER FILE ON THE DISK.

161 002145 047 01 0 000200 000200
162 002146 047 01 0 000200 000200
163 002147 047 01 0 000200 000200
164 002148 047 01 0 000200 000200
165 002149 047 01 0 000200 000200
166 002150 047 01 0 000200 000200
167 002151 047 01 0 000200 000200
168 002152 047 01 0 000200 000200
169 002153 047 01 0 000200 000200
170 002154 047 01 0 000200 000200
171 002155 047 01 0 000200 000200
172 002156 047 01 0 000200 000200
173 002157 047 01 0 000200 000200
174 002158 047 01 0 000200 000200
175 002159 047 01 0 000200 000200
176 002160 047 01 0 000200 000200

IFIND SUITABLE NAME FOR PARAMETER FILE

200 002161 047 01 0 000200 000200
201 002162 047 01 0 000200 000200
202 002163 047 01 0 000200 000200
203 002164 047 01 0 000200 000200
204 002165 047 01 0 000200 000200
205 002166 047 01 0 000200 000200
206 002167 047 01 0 000200 000200
207 002168 047 01 0 000200 000200
208 002169 047 01 0 000200 000200
209 002170 047 01 0 000200 000200
210 002171 047 01 0 000200 000200
211 002172 047 01 0 000200 000200
212 002173 047 01 0 000200 000200
213 002174 047 01 0 000200 000200
LPRINT MACRO 47(284)-9 08129 2-AUG-74 PAGE 1-4

213 00179 603 00 00 77006R
214 0017A 254 00 00 000172
215 0017B 258 00 00 000155
216 0027A 252 00 00 000025
217 00279 208 00 00 000001
218 00278 076 00 00 000022
219 00277 625 33 00 77777
220 00276 254 00 00 00166
221 00275 326 00 00 00166
222 00274 260 32 00 000025
223 00273 225 03 00 616545
224 0022B 200 02 00 000024
225 000024 200 05 00 000011
226 000024 378 00 00 000000
227 000024 277 00 00 000000
228 000024 254 04 00 000000
229 000024 200 01 00 000254
230 000024 202 31 00 000011
231 000024 402 02 00 000012
232 000024 207 02 00 000011
233 000024 270 00 00 000000
234 000024 071 00 00 000000
235 000024 400 02 00 000000
236 000024 291 00 00 000000
237 000024 291 00 00 000000
238 000024 209 00 00 000000
239 000024 350 00 00 000000
240 000024 314 00 00 000000
241 000024 546 00 00 000000
242 000024 247 00 00 000000
243 000024 047 00 00 000000
244 000024 252 00 00 000000
245 000024 253 00 00 000000
246 000024 264 00 00 000000
247 000024 002 00 00 000000
248 000024 002 00 00 000000
249 000024 004 00 00 000000
250 000024 004 00 00 000000
251 000024 004 00 00 000000
252 000024 004 00 00 000000
253 000024 004 00 00 000000
254 000024 004 00 00 000000

NO ERRORS DETECTED

PROGRAM BREAK IS 008256

3K CORE USED
TITLE: WRTED

COMMENT:

WRITTEN BY GARRET A. VANDER LUST COMPLETED 30-MAY-73

PART OF LEDIT.SHR; VERSION 3

WRTED.MAC PERFORMS ALL OF THE LESSON EDITING FUNCTIONS FOR THE LEARN SYSTEM.

CONTENTS OF WRTED.MAC

SECTION A. CONTAINS ENTRY POINTS TO WRTED.MAC

SECTION B. INTERPRETS 1ST. CHARACTER IN AN EDIT COMMAND.

SECTION C. EXECUTES "X" COMMANDS.

SECTION D. EXECUTES "A" COMMANDS.

SECTION E. EXECUTES "T" COMMANDS.

SECTION F. EXECUTES "D" COMMANDS.

SECTION G. EXECUTES "I" COMMANDS.

SECTION H. EXECUTES "R" COMMANDS.

SECTION I. EXECUTES "K" COMMANDS.

SECTION J. EXECUTES "C" AND "N" COMMANDS.


SECTION L. SUBROUTINE TO INITIALIZE SPACE LIST.

SECTION M. SUBROUTINE TO FILL EDITING BUFFER.

SECTION N. SUBROUTINE TO EMPTY EDITING BUFFER.

SECTION O. SUBROUTINE TO RETURN TO BEGINNING OF LESSON.

SECTION P. SUBROUTINE TO FIND FRAME TO BE EDITED.

GENERAL DESCRIPTION OF LEARN LESSON EDITOR
THE LEARN LESSON EDITOR CONTAINS THREE MAJOR COMPONENTS: AN
EDITING BUFFER, A SPACE LIST, AND A LINK LIST. A FRAME WHICH IS TO
BE EDITED IS PLACED IN THE EDITING BUFFER WHICH CONSISTS OF 64 SLOTS.
EACH OF WHICH OCCUPIES 16 WORDS OF CORE, THE SLOTS ARE NUMBERED
FROM 0 TO 63, AND EACH LINE IN THE FRAME IS PLACED IN A SEPARATE SLOT.
THE SPACE LIST IS A LIST OF SLOTS WHICH ARE EMPTY, WHEN A LINE IS
DELETED, THE NO. OF THE SLOT WHICH CONTAINED IT IS PLACED ON TOP
OF THE SPACE LIST, AND WHEN A LINE IS INSERTED, THE NO. OF THE SLOT
INTO WHICH IT IS PLACED IS REMOVED FROM THE TOP OF THE SPACE LIST.
AFTER DELETIONS AND INSERTIONS, THE LINES IN THE BUFFER MAY NO LONGER
BE IN ORDER OF INCREASING LINE NO., FOR THIS REASON THE ORDER OF THE
LINES IS STORED IN THE LINK LIST, EACH WORD IN THE LINK LIST COR-
RESPONDS TO A SLOT IN THE EDITING BUFFER, THE CONTENT OF EACH WORD
IN THE LINK LIST IS THE SLOT NO. OF THE SLOT CONTAINING THE NEXT LINE
IN THE FRAME. FOR EXAMPLE, IF THE 5TH LINE IS IN SLOT 7 AND THE 6TH
LINE IS IN SLOT 4, THEN THE 8TH WORD IN THE LINK LIST WILL CONTAIN
THE NUMBER 4. (WORDS IN THE LINK LIST ARE NUMBERED FROM 0 TO 63.) A
MINUS ONE IS PLACED IN THE WORD IN THE LINK LIST WHICH CORRESPONDS
TO THE SLOT IN THE EDITING BUFFER WHICH CONTAINS THE LAST LINE IN THE
FRAME. THE FIRST LINE IN A FRAME IS ALWAYS PlACED IN SLOT 2, AND
IT ALWAYS REMAINS THERE SINCE NO LINES MAY BE INSERTED AHEAD OF IT.

IN ORDER TO DELETE A LINE IT IS ONLY NECESSARY TO CHANGE THE
LINK LIST, THE EXAMPLE BELOW SHOWS HOW LINE 3 CAN BE DELETED.
AFTER DELETION THE POINTER TO SLOT 3 IN THE
EDITING BUFFER IT NO LONGER POINTS TO SLOT 2, SLOT 2 WHICH CONT-
AINED THE THIRD LINE IN THE FRAME, WOULD BE PLACED ON TOP OF THE
SPACE LIST.

BEFORE DELETION       AFTER DELETION

|  LINK*01 |  1 |  LINK*01 |  1 |
|  LINK*11 |  2 |  LINK*11 |  3 |
|  LINK*21 |  3 |  LINK*21 |  3 |
|  LINK*31 |  4 |  LINK*31 |  4 |
|  LINK*41 | -1 |  LINK*41 | -1 |

NOTICE THAT NO CHARACTERS ARE ACTUALLY REMOVED FROM THE EDITING BUFFER.

THE EXAMPLE BELOW SHOWS THE CHANGE IN THE LINK LIST WHICH WOULD
OCUR IF A LINE WAS INSERTED IN FRONT OF LINE 2.

BEFORE INSERTION       AFTER INSERTION

|  LINK*01 |  1 |  LINK*01 |  5 |
|  LINK*11 |  2 |  LINK*11 |  2 |
|  LINK*21 |  3 |  LINK*21 |  3 |
|  LINK*31 |  4 |  LINK*31 |  4 |
|  LINK*41 | -1 |  LINK*41 | -1 |

REPLACING LINES DOES NOT AFFECT THE LINK LIST.
NOTE: THE LESSON EDITOR PLACES AN ASCII CHARACTER WITH OCTAL CODE 36 AT THE BEGINNING OF EACH FRAME. THE ASCII CHARACTER WITH OCTAL CODE 84 IS PLACED AT THE END OF THE LESSON. FRAMES MAY BE NUMBERED FROM 1 THRU 999, AND THERE MAY BE NO MORE THAN 64 LINES IN A FRAME.

ENTRY POINTS TO WRTE0,MAC (ALL IN SECTION A)

EDIT2 = FROM LEDIT,MAC (SECTION I) IN RESPONSE TO USER'S IE COMMAND
WHILE IN TEST MODE (SEE INTERP,MAC)

EDIT = FROM LEDIT,MAC (SECTION I) IN RESPONSE TO USER'S EDIT COMMAND

WRIT = FROM LEDIT,MAC (SECTION I) IN RESPONSE TO USER'S WRITE COMMAND

EXTERNAL SUBROUTINES CALLED BY WRTE0,MAC

EOLIN = INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE END OF A LINE IS REACHED, LOCATED IN AUX2,MAC (SECTION A)

IID = INPUTS AN INTEGER FROM THE DISK AND PLACES IT IN AC-B, LOCATED IN AUX2,MAC (SECTION I)

IIT = INPUTS AN INTEGER FROM THE TELETYPewriter AND PLACES IT IN AC-B, LOCATED IN AUX2,MAC (SECTION M)

IIT2 = INPUTS AN INTEGER FROM THE TELETYPewriter AND PLACES IT IN AC-B, THE 1ST Digit IS ASSUMED TO BE IN AC-A, LOCATED IN AUX2,MAC (SECTION M)

INCHR = INPUTS A CHARACTER FROM THE DISK AND PLACES IT IN AC-A, LOCATED IN AUX2,MAC (SECTION B)

IOD = OUTPUTS AN INTEGER IN AC-A TO THE DISK, LOCATED IN AUX2,MAC (SECTION E)

IOT = OUTPUTS AN INTEGER IN AC-A TO THE TELETYPewriter, LOCATED IN AUX2,MAC (SECTION G)

OUTCHR = OUTPUTS A CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX2,MAC (SECTION G)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LDTLOW,MAC)

ARGMH = ARGUMENTS FOR GETSEG UUO (6 WORDS)
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>BUFFER - LESSON EDITING BUFFER (1024 WORDS)</td>
</tr>
<tr>
<td>161</td>
<td>FRAME - CONTAINS FRAME INCREMENT NO.</td>
</tr>
<tr>
<td>164</td>
<td>FRMNUM - CONTAINS NO. OF FRAME CURRENTLY IN THE EDITING BUFFER</td>
</tr>
<tr>
<td>165</td>
<td>FRMTYP - CONTAINS LETTER INDICATING TYPE OF FRAME IN THE EDITING BUFFER</td>
</tr>
<tr>
<td>169</td>
<td>1ST FRAME NO. IN EDIT COMMAND IS STORED HERE</td>
</tr>
<tr>
<td>170</td>
<td>FSTLIN - 1ST LINE NO. IN EDIT COMMAND IS STORED HERE</td>
</tr>
<tr>
<td>174</td>
<td>IBUF - BUFFER HEADER FOR DISK INPUT (3 WORDS)</td>
</tr>
<tr>
<td>175</td>
<td>INAM - LOOKUP ARGUMENTS FOR DISK INPUT (4 WORDS)</td>
</tr>
<tr>
<td>176</td>
<td>INBUF - DISK INPUT BUFFER NO. 1 (131 WORDS)</td>
</tr>
<tr>
<td>179</td>
<td>INBUF2 - DISK INPUT BUFFER NO. 2 (131 WORDS)</td>
</tr>
<tr>
<td>182</td>
<td>LINESUB - LINE CONTAINING A STRING TO BE REPLACED IS STORED HERE (16 WORDS)</td>
</tr>
<tr>
<td>184</td>
<td>LINK - LINK LIST (64 WORDS)</td>
</tr>
<tr>
<td>187</td>
<td>LSTFRM - 1ST FRAME NO. IN EDIT COMMAND IS STORED HERE</td>
</tr>
<tr>
<td>189</td>
<td>LSTLIN - 1ST LINE NO. IN EDIT COMMAND IS STORED HERE</td>
</tr>
<tr>
<td>190</td>
<td>NXTFRM - CONTAINS NO. OF FRAME FOLLOWING FRAME IN THE EDITING BUFFER</td>
</tr>
<tr>
<td>193</td>
<td>OBUF - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)</td>
</tr>
<tr>
<td>195</td>
<td>ODR - STRING WHICH IS TO BE REPLACED IS STORED HERE (3 WORDS)</td>
</tr>
<tr>
<td>199</td>
<td>ONAM - ENTER ARGUMENTS FOR DISK OUTPUT (4 WORDS)</td>
</tr>
<tr>
<td>200</td>
<td>OUTF2 - DISK OUTPUT BUFFER NO. 2 (131 WORDS)</td>
</tr>
<tr>
<td>202</td>
<td>OUTF2 - DISK OUTPUT BUFFER NO. 1 (131 WORDS)</td>
</tr>
<tr>
<td>205</td>
<td>PC1 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>206</td>
<td>PC2 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>209</td>
<td>PC3 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>210</td>
<td>PC4 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>212</td>
<td>POLST - PUSH DOWN LIST USED TO STORE ASCII DIGITS OF A</td>
</tr>
</tbody>
</table>
NUMBER BEING CONVERTED FROM BINARY TO ASCII CODE (8 WORDS)

SPACE = SPACE LIST (64 WORDS)
STORE = COMMAND INDEX IS PLACED HERE SO THAT COMMANDS MAY BE PASSED FROM ONE HIGH SEGMENT TO ANOTHER

TRMLK = TRMOP, ARGUMENTS (3 WORDS)

DESCRIPTION OF FLAGS STORED IN AC-8

<table>
<thead>
<tr>
<th>BIT</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>COPY</td>
</tr>
<tr>
<td>17</td>
<td>NO EMPTY</td>
</tr>
<tr>
<td>23</td>
<td>APPEND INSIDE OF FRAME</td>
</tr>
<tr>
<td>24</td>
<td>NEW FRAME NO.</td>
</tr>
<tr>
<td>25</td>
<td>OLD FRAME NO.</td>
</tr>
<tr>
<td>26</td>
<td>LAST FRAME</td>
</tr>
<tr>
<td>27</td>
<td>QUIT</td>
</tr>
<tr>
<td>29</td>
<td>OLD LESSON</td>
</tr>
<tr>
<td>34</td>
<td>TEST</td>
</tr>
<tr>
<td>39</td>
<td>PROCESS</td>
</tr>
</tbody>
</table>

% WISEG

ENTRY_HRT;EDT;EDT2

EXTERN BUFFER,NXTFRM,STORE,FRMING,ARGMNT,IOD,IBUF,OBUF
EXTERN FRMNUM,FRMTYP,IOT,PODST,INCMR,OUTCR,KDLIN
EXTERN PC1,ONAM,INAM,LINK,PC2,SPACE,PC3,PC4,TRMLK
EXTERN STFRM,LSTFRM,FSTLIN,LSTLIN,OLDSTR,LINBUF
EXTERN IT1,IT2,IT3,INBUF,OUTBUF,INBUF,OUTBUF

FLAGS=0

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>A=1</td>
</tr>
<tr>
<td>00002</td>
<td>B=2</td>
</tr>
<tr>
<td>00003</td>
<td>C=3</td>
</tr>
<tr>
<td>00004</td>
<td>D=4</td>
</tr>
<tr>
<td>00005</td>
<td>E=5</td>
</tr>
<tr>
<td>00006</td>
<td>F=6</td>
</tr>
<tr>
<td>00007</td>
<td>G=7</td>
</tr>
<tr>
<td>00010</td>
<td>H=10</td>
</tr>
<tr>
<td>00011</td>
<td>I=11</td>
</tr>
<tr>
<td>00012</td>
<td>J=12</td>
</tr>
<tr>
<td>00017</td>
<td>Q=17</td>
</tr>
</tbody>
</table>

PROTI: 17788
POINT1: POINT 6,INAM
SECTION A, INPUT LESSON NAME AND FRAME INCREMENT NUMBER, IF ANY.
THE LOW SEGMENT IS EXPENDED TO 2K. BUFFERS ARE ADDED TO THE DISK
BUFFER RINGS. FRNUM AND NTXNUM ARE SET TO ZERO, AND THE LAST
FRAME FLAG IS CLEARED.

EDT21: MOVEL 0,PC1  ;SAVE RETURN ADDRESS
TRO FLAGS,100  ;SET OLD LESSON FLAG
MOVEL 0,12
MOVEL B,FRMINC
;INITIALIZE FRAME INCREMENT TO 10
JRTS
WRTI TRA2 FLAGS,100  ;CLEAR OLD LESSON FLAG AND SKIP NEXT LINE
EDT1 TRO FLAGS,100  ;SET OLD LESSON FLAG
MOVEL 0,PC1  ;SAVE RETURN ADDRESS
MOVEL 0,12
MOVEL B,FRMINC
;INITIALIZE FRAME INCREMENT TO 10
TTCALL A,4  ;INPUT NEXT CHARACTER IN COMMAND STRING
CAIN A,40  ;IS CHARACTER A SPACE?
JRTS +2  ;YES, GO INPUT NEXT CHARACTER
CAINE A,12
CAI A,71
JST NONAME  ;IF END OF COMMAND STRING, GO TO NONAME
TCALL A,60
CAI A,71
JST NONUM  ;IF CHARACTER IS NOT A NUMBER, GO TO NONUM
TCALL A,11
TTCALL 3,ASC12  ;LES$ON NAMES MAY NOT BEGIN WITH A NUMERICAL
/)

JRTS 9PC1  ;RETURN TO EDIT SUBPROGRAM
NONUNH SETEB C,INAM
MOVE B,POINC
JST +2  ;FIRST CHARACTER OF LESSON NAME IS ALREADY IN
TTCALL 4,A  ;INPUT CHARACTER
CAIN A,40  ;IS CHARACTER A SPACE?
JRTS 112
CAINE A,15
JST NAMINC
CAIN A,15
JST NAMINC
CAINE A,15
JST NAMINC
CAI A,"\""
JST SWITHCH  ;YES, GO TO SWITCH
CAIN C,0
JST 356 03  ;IS THIS 7TH CHARACTER IN LESSON NAME?
ADDI A,40  ;ICONVERT TO SIXBIT CODE
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>319</td>
<td>002053</td>
<td>MOV AL, 136</td>
</tr>
<tr>
<td>320</td>
<td>002054</td>
<td>MOV AL, 31</td>
</tr>
<tr>
<td>321</td>
<td>002055</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>322</td>
<td>002056</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>323</td>
<td>002057</td>
<td>CMP AL, 01</td>
</tr>
<tr>
<td>324</td>
<td>002058</td>
<td>JNZ 421H</td>
</tr>
<tr>
<td>325</td>
<td>002059</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>326</td>
<td>00205A</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>327</td>
<td>00205B</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>328</td>
<td>00205C</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>329</td>
<td>00205D</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>330</td>
<td>00205E</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>331</td>
<td>00205F</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>332</td>
<td>002060</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>333</td>
<td>002061</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>334</td>
<td>002062</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>335</td>
<td>002063</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>336</td>
<td>002064</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>337</td>
<td>002065</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>338</td>
<td>002066</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>339</td>
<td>002067</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>340</td>
<td>002068</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>341</td>
<td>002069</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>342</td>
<td>00206A</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>343</td>
<td>00206B</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>344</td>
<td>00206C</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>345</td>
<td>00206D</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>346</td>
<td>00206E</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>347</td>
<td>00206F</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>348</td>
<td>002070</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>349</td>
<td>002071</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>350</td>
<td>002072</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>351</td>
<td>002073</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>352</td>
<td>002074</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>353</td>
<td>002075</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>354</td>
<td>002076</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>355</td>
<td>002077</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>356</td>
<td>002078</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>357</td>
<td>002079</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>358</td>
<td>00207A</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>359</td>
<td>00207B</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>360</td>
<td>00207C</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>361</td>
<td>00207D</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>362</td>
<td>00207E</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>363</td>
<td>00207F</td>
<td>CMP AL, 00</td>
</tr>
<tr>
<td>364</td>
<td>002080</td>
<td>JZ 442H</td>
</tr>
<tr>
<td>365</td>
<td>002081</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>366</td>
<td>002082</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>367</td>
<td>002083</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>368</td>
<td>002084</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>369</td>
<td>002085</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>370</td>
<td>002086</td>
<td>MOV AL, 00</td>
</tr>
<tr>
<td>371</td>
<td>002087</td>
<td>MOV AL, 00</td>
</tr>
</tbody>
</table>

**Notes:**
- The code appears to be a machine language program, likely part of a larger software system.
- The program seems to be dealing with character manipulation and frame handling.
- The instructions are in hexadecimal format, indicating assembly language programming.
- The program transitions between different instructions based on the comparison of AL register values.

**Possible Uses:**
- This code could be part of a terminal or communication software.
- The program might be used for data validation or error handling.
- The code structure suggests it's designed for specific hardware and system requirements.
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Label</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>372</td>
<td>001241</td>
<td><strong>CORE A</strong></td>
<td><strong>EXPAND LOW SEGMENT TO 2K</strong></td>
</tr>
<tr>
<td>373</td>
<td>001292</td>
<td><strong>MALT</strong></td>
<td></td>
</tr>
<tr>
<td>374</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>375</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>376</td>
<td>001261</td>
<td><strong>IADD INPUT AND OUTPUT BUFFERS TO DISK BUFFER RINGS</strong></td>
<td></td>
</tr>
<tr>
<td>377</td>
<td>001271</td>
<td><strong>SET2 INBUF2</strong></td>
<td></td>
</tr>
<tr>
<td>378</td>
<td>001301</td>
<td><strong>MOVE A, INBUF2</strong></td>
<td></td>
</tr>
<tr>
<td>379</td>
<td>001311</td>
<td><strong>MOVE A, INBUF1</strong></td>
<td></td>
</tr>
<tr>
<td>380</td>
<td>001321</td>
<td><strong>MOVE A, INBUF2</strong></td>
<td></td>
</tr>
<tr>
<td>381</td>
<td>001331</td>
<td><strong>MOVE A, INBUF1</strong></td>
<td></td>
</tr>
<tr>
<td>382</td>
<td>001341</td>
<td><strong>MOVE A, OUTBUF2</strong></td>
<td></td>
</tr>
<tr>
<td>383</td>
<td>001351</td>
<td><strong>MOVE A, OUTBUF1</strong></td>
<td></td>
</tr>
<tr>
<td>384</td>
<td>001361</td>
<td><strong>MOVE A, OUTBUF2</strong></td>
<td></td>
</tr>
<tr>
<td>385</td>
<td>001371</td>
<td><strong>MOVE A, OUTBUF1</strong></td>
<td></td>
</tr>
<tr>
<td>386</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>387</td>
<td>001401</td>
<td><strong>MOVSI A, UNP</strong></td>
<td></td>
</tr>
<tr>
<td>388</td>
<td>001411</td>
<td><strong>MOVEM A, INAM</strong></td>
<td></td>
</tr>
<tr>
<td>389</td>
<td>001421</td>
<td><strong>MOVEM A, INAM</strong></td>
<td></td>
</tr>
<tr>
<td>390</td>
<td>001431</td>
<td><strong>MOVEM A, INAM</strong></td>
<td></td>
</tr>
<tr>
<td>391</td>
<td>001441</td>
<td><strong>LOOKUP A, INAM</strong></td>
<td></td>
</tr>
<tr>
<td>392</td>
<td>001451</td>
<td><strong>LOOKUP LESSON FILE</strong></td>
<td></td>
</tr>
<tr>
<td>393</td>
<td>001461</td>
<td><strong>JMP NTND</strong></td>
<td></td>
</tr>
<tr>
<td>394</td>
<td>001471</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>395</td>
<td>001481</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>396</td>
<td>001491</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>397</td>
<td>001501</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>398</td>
<td>001511</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>399</td>
<td>001521</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>001531</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>001541</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>001551</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>403</td>
<td>001561</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>404</td>
<td>001571</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>405</td>
<td>001581</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>406</td>
<td>001591</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>407</td>
<td>001601</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>408</td>
<td>001611</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>409</td>
<td>001621</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>410</td>
<td>001631</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>411</td>
<td>001641</td>
<td><strong>TRNE FLRS, 104</strong></td>
<td></td>
</tr>
<tr>
<td>412</td>
<td>001651</td>
<td><strong>TRNE FLRS, 120</strong></td>
<td></td>
</tr>
<tr>
<td>413</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>414</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>415</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>416</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>417</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>418</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>419</td>
<td>001721</td>
<td><strong>GETCHAR</strong></td>
<td></td>
</tr>
<tr>
<td>420</td>
<td>001731</td>
<td><strong>MOVE A, TRMPL</strong></td>
<td></td>
</tr>
<tr>
<td>421</td>
<td>001741</td>
<td><strong>TRMPL, A</strong></td>
<td></td>
</tr>
<tr>
<td>422</td>
<td>001751</td>
<td><strong>MOVE C, 1</strong></td>
<td></td>
</tr>
<tr>
<td>423</td>
<td>001761</td>
<td><strong>SLEEP C</strong></td>
<td></td>
</tr>
<tr>
<td>424</td>
<td>001771</td>
<td><strong>ISTDOUT</strong></td>
<td></td>
</tr>
</tbody>
</table>

**SECTION B. INPUT FIRST CHARACTER IN EDIT COMMAND STRING AND BRANCH TO THE PROPER ROUTINE TO EXECUTE THE COMMAND.**

**GETCHAR**

**MOVE A, TRMPL**

**MOVE ADDRESS OF TRMPL, ARGUMENTS TO AC-A**

**ISkip IF TTY OUTPUT BUFFER IS NOT EMPTY**

**ISLEEP A SEC BEFORE CHECKING BUFFER AGAIN**
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN</td>
<td>Append lines onto frame N</td>
</tr>
<tr>
<td>CN=M</td>
<td>Copy frame N and number it M</td>
</tr>
<tr>
<td>DN</td>
<td>Delete frame N</td>
</tr>
<tr>
<td>DN=M</td>
<td>Delete frames N thru M</td>
</tr>
<tr>
<td>DN,P</td>
<td>Delete line P in frame N</td>
</tr>
<tr>
<td>DN,P-Q</td>
<td>Delete lines P thru Q in frame N</td>
</tr>
</tbody>
</table>
**TABLE 1-11**

<table>
<thead>
<tr>
<th>Frame N</th>
<th>Frame P</th>
<th>Frame Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line 1</td>
<td>Replace</td>
<td>Replace</td>
</tr>
<tr>
<td>Line 2</td>
<td>Frame N</td>
<td>Lines P-Q</td>
</tr>
</tbody>
</table>
SECTION C. EXECUTE "X" COMMAND. STORE FRAME NUMBER INCREMENT IN FRMING.

<table>
<thead>
<tr>
<th>Line</th>
<th>Hexadecimal</th>
<th>Binary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>516</td>
<td>000011</td>
<td>101111</td>
<td>XNUM: JSP Q.111T INPUT INCREMENT NO.</td>
</tr>
<tr>
<td>519</td>
<td>000012</td>
<td>101110</td>
<td>JUMPE B.XERR IF NO. IS ZERO, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>520</td>
<td>000013</td>
<td>101110</td>
<td>JST .+2 GO CHECK 1ST CHARACTER AFTER NO.</td>
</tr>
<tr>
<td>521</td>
<td>000014</td>
<td>101110</td>
<td>TDCALL 4.A IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>522</td>
<td>000015</td>
<td>101110</td>
<td>CAIN A.40</td>
</tr>
<tr>
<td>523</td>
<td>000016</td>
<td>101110</td>
<td>JST .-2 YES. GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>524</td>
<td>000017</td>
<td>101110</td>
<td>CALE A.15 END OF COMMAND STRING?</td>
</tr>
<tr>
<td>525</td>
<td>000018</td>
<td>101110</td>
<td>JST ILMCHD NO. GO TYPE ERROR MESSAGE (SECTION B)</td>
</tr>
<tr>
<td>526</td>
<td>000019</td>
<td>101110</td>
<td>MOVE B.FRINFO STORE INCREMENT IN FRMING</td>
</tr>
<tr>
<td>527</td>
<td>000020</td>
<td>101110</td>
<td>TDCALL 4.A INPUT LINE FEED</td>
</tr>
<tr>
<td>528</td>
<td>000021</td>
<td>101110</td>
<td>JST GETCHD IGO ACCEPT NEXT COMMAND (SECTION B)</td>
</tr>
<tr>
<td>529</td>
<td>000022</td>
<td>101110</td>
<td>TDCALL 11.0</td>
</tr>
</tbody>
</table>

SECTION D. EXECUTE "A" COMMAND. "A" COMMANDS ARE USED TO ADD LINES TO THE END OF A FRAME OR FRAMES TO THE END OF A LESSON.

<table>
<thead>
<tr>
<th>Line</th>
<th>Hexadecimal</th>
<th>Binary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>530</td>
<td>000011</td>
<td>101111</td>
<td>APPND: TDCALL 4.A INPUT NEXT CHARACTER IN COMMAND STRING</td>
</tr>
<tr>
<td>531</td>
<td>000012</td>
<td>101110</td>
<td>CAIN A.40 IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>532</td>
<td>000013</td>
<td>101110</td>
<td>JST .-2 YES. GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>533</td>
<td>000014</td>
<td>101110</td>
<td>CAIN A.15 END OF COMMAND STRING?</td>
</tr>
<tr>
<td>534</td>
<td>000015</td>
<td>101110</td>
<td>JST APPALL YES. GO TO APPALL</td>
</tr>
<tr>
<td>535</td>
<td>000016</td>
<td>101110</td>
<td>JST XERR INPUT FRMNO.</td>
</tr>
<tr>
<td>536</td>
<td>000017</td>
<td>101110</td>
<td>TDCALL 4.A IGO CHECK 1ST CHARACTER AFTER NO.</td>
</tr>
<tr>
<td>537</td>
<td>000018</td>
<td>101110</td>
<td>CAIN A.40 IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>538</td>
<td>000019</td>
<td>101110</td>
<td>JST .+2 YES. GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>539</td>
<td>000020</td>
<td>101110</td>
<td>CALE A.15 END OF COMMAND STRING?</td>
</tr>
<tr>
<td>540</td>
<td>000021</td>
<td>101110</td>
<td>JST ILMCHD NO. GO TYPE ERROR MESSAGE (SECTION B)</td>
</tr>
<tr>
<td>541</td>
<td>000022</td>
<td>101110</td>
<td>TDCALL 4.A INPUT LINE FEED</td>
</tr>
<tr>
<td>542</td>
<td>000023</td>
<td>101110</td>
<td>MOVE B.FRINFO STORE FRMNO. IN FRINFO</td>
</tr>
<tr>
<td>543</td>
<td>000024</td>
<td>101110</td>
<td>JSP Q.FRINFO FIND FRAME TO WHICH LINES WILL BE APPENDED (SECTION P)</td>
</tr>
<tr>
<td>544</td>
<td>000025</td>
<td>101110</td>
<td>TRO FLAGS.10000 SET APPEND INSIDE OF FRAME FLAG</td>
</tr>
<tr>
<td>Line</td>
<td>Address</td>
<td>Contents</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>001008'</td>
<td>001008'</td>
<td>JUMP A.,-3</td>
<td></td>
</tr>
<tr>
<td>001009'</td>
<td>001009'</td>
<td>POP D.,A</td>
<td></td>
</tr>
<tr>
<td>00100A'</td>
<td>00100A'</td>
<td>IGET CHARACTER FROM PUSH DOWN LIST</td>
<td></td>
</tr>
<tr>
<td>00100B'</td>
<td>00100B'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>00100C'</td>
<td>00100C'</td>
<td>IPLACE CHARACTER IN THE EDITING LIST</td>
<td></td>
</tr>
<tr>
<td>00100D'</td>
<td>00100D'</td>
<td>MLRE C,D</td>
<td></td>
</tr>
<tr>
<td>00100E'</td>
<td>00100E'</td>
<td>IGET NO. OF CHARACTERS REMAINING IN AC-G</td>
<td></td>
</tr>
<tr>
<td>00100F'</td>
<td>00100F'</td>
<td>JUMP C.,-3</td>
<td></td>
</tr>
<tr>
<td>001010'</td>
<td>001010'</td>
<td>MOVEI A.,0</td>
<td></td>
</tr>
<tr>
<td>001011'</td>
<td>001011'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001012'</td>
<td>001012'</td>
<td>IPUT SPACE AFTER NO. IN EDITING BUFFER</td>
<td></td>
</tr>
<tr>
<td>001013'</td>
<td>001013'</td>
<td>IPUT REST OF FIRST LINE IN EDITING BUFFER</td>
<td></td>
</tr>
<tr>
<td>001014'</td>
<td>001014'</td>
<td>MOVEI A.,&quot;Y&quot;</td>
<td></td>
</tr>
<tr>
<td>001015'</td>
<td>001015'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001016'</td>
<td>001016'</td>
<td>MOVEI A.,&quot;Y&quot;</td>
<td></td>
</tr>
<tr>
<td>001017'</td>
<td>001017'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001018'</td>
<td>001018'</td>
<td>MOVEI A.,&quot;p&quot;</td>
<td></td>
</tr>
<tr>
<td>001019'</td>
<td>001019'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>00101A'</td>
<td>00101A'</td>
<td>MOVEI A.,&quot;g&quot;</td>
<td></td>
</tr>
<tr>
<td>00101B'</td>
<td>00101B'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>00101C'</td>
<td>00101C'</td>
<td>MOVEI A.,&quot;m&quot;</td>
<td></td>
</tr>
<tr>
<td>00101D'</td>
<td>00101D'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>00101E'</td>
<td>00101E'</td>
<td>MOVE A.,FRMTYP</td>
<td></td>
</tr>
<tr>
<td>00101F'</td>
<td>00101F'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001020'</td>
<td>001020'</td>
<td>MOVEI A.,15</td>
<td></td>
</tr>
<tr>
<td>001021'</td>
<td>001021'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001022'</td>
<td>001022'</td>
<td>MOVEI A.,12</td>
<td></td>
</tr>
<tr>
<td>001023'</td>
<td>001023'</td>
<td>IDPB A.,G</td>
<td></td>
</tr>
<tr>
<td>001024'</td>
<td>001024'</td>
<td>MOVEI E.,2</td>
<td></td>
</tr>
<tr>
<td>001025'</td>
<td>001025'</td>
<td>ISET LINE NO. TO 2</td>
<td></td>
</tr>
<tr>
<td>001026'</td>
<td>001026'</td>
<td>NxTLINE</td>
<td></td>
</tr>
<tr>
<td>001027'</td>
<td>001027'</td>
<td>CAIN E.,67</td>
<td></td>
</tr>
<tr>
<td>001028'</td>
<td>001028'</td>
<td>IIS THIS THE 55TH LINE?</td>
<td></td>
</tr>
<tr>
<td>001029'</td>
<td>001029'</td>
<td>JIRST WARN</td>
<td></td>
</tr>
<tr>
<td>00102A'</td>
<td>00102A'</td>
<td>IYES, GO TYPE WARNING</td>
<td></td>
</tr>
<tr>
<td>00102B'</td>
<td>00102B'</td>
<td>CALL E.,101</td>
<td></td>
</tr>
<tr>
<td>00102C'</td>
<td>00102C'</td>
<td>IIS THIS THE 56TH LINE?</td>
<td></td>
</tr>
<tr>
<td>00102D'</td>
<td>00102D'</td>
<td>JIRST BUFFUL</td>
<td></td>
</tr>
<tr>
<td>00102E'</td>
<td>00102E'</td>
<td>IYES, GO TYPE ERROR MESSAGE</td>
<td></td>
</tr>
<tr>
<td>00102F'</td>
<td>00102F'</td>
<td>WRNDONI</td>
<td></td>
</tr>
<tr>
<td>001030'</td>
<td>001030'</td>
<td>MOVEI A.,40</td>
<td></td>
</tr>
<tr>
<td>001031'</td>
<td>001031'</td>
<td>CAIGE E.,12</td>
<td></td>
</tr>
<tr>
<td>001032'</td>
<td>001032'</td>
<td>TTECALL 1,A</td>
<td></td>
</tr>
<tr>
<td>001033'</td>
<td>001033'</td>
<td>ITYPE SPACE BEFORE LINE NO. LESS THAN 10</td>
<td></td>
</tr>
<tr>
<td>001034'</td>
<td>001034'</td>
<td>MOVE A.,E,5</td>
<td></td>
</tr>
<tr>
<td>001035'</td>
<td>001035'</td>
<td>JSP 0,10T</td>
<td></td>
</tr>
<tr>
<td>001036'</td>
<td>001036'</td>
<td>ITYPE LINE NO.</td>
<td></td>
</tr>
<tr>
<td>001037'</td>
<td>001037'</td>
<td>TTECALL 3,SPCAST</td>
<td></td>
</tr>
<tr>
<td>001038'</td>
<td>001038'</td>
<td>TTECALL 4,A</td>
<td></td>
</tr>
<tr>
<td>001039'</td>
<td>001039'</td>
<td>IINPUT 1ST CHARACTER OF LINE</td>
<td></td>
</tr>
<tr>
<td>00103A'</td>
<td>00103A'</td>
<td>CAIGE A.,33</td>
<td></td>
</tr>
<tr>
<td>00103B'</td>
<td>00103B'</td>
<td>IIS CHARACTER AN &quot;ESC&quot;?</td>
<td></td>
</tr>
<tr>
<td>00103C'</td>
<td>00103C'</td>
<td>CAIN E.,75</td>
<td></td>
</tr>
<tr>
<td>00103D'</td>
<td>00103D'</td>
<td>JIRST STOPF</td>
<td></td>
</tr>
<tr>
<td>00103E'</td>
<td>00103E'</td>
<td>IYES, GO TO STOPF</td>
<td></td>
</tr>
<tr>
<td>00103F'</td>
<td>00103F'</td>
<td>POP J,G</td>
<td></td>
</tr>
<tr>
<td>001040'</td>
<td>001040'</td>
<td>IGET OPEN SLOT FROM SPACE LIST</td>
<td></td>
</tr>
<tr>
<td>001041'</td>
<td>001041'</td>
<td>MOVEH G, LINK F</td>
<td></td>
</tr>
<tr>
<td>001042'</td>
<td>001042'</td>
<td>IPUT SLOT NO. IN LINK LIST</td>
<td></td>
</tr>
<tr>
<td>001043'</td>
<td>001043'</td>
<td>MOVE F,G</td>
<td></td>
</tr>
<tr>
<td>001044'</td>
<td>001044'</td>
<td>ISAVE SLOT NO. IN AC-F</td>
<td></td>
</tr>
<tr>
<td>001045'</td>
<td>001045'</td>
<td>ISET UP BYTE POINTER TO PROPER SLOT</td>
<td></td>
</tr>
<tr>
<td>001046'</td>
<td>001046'</td>
<td>INULI G,20</td>
<td></td>
</tr>
<tr>
<td>001047'</td>
<td>001047'</td>
<td>IEACH SLOT OCCUPIES 16 WORDS OF CORE</td>
<td></td>
</tr>
<tr>
<td>001048'</td>
<td>001048'</td>
<td>ADDI G,BUFFER</td>
<td></td>
</tr>
<tr>
<td>001049'</td>
<td>001049'</td>
<td>IADD ON BASE ADDRESS OF BUFFER</td>
<td></td>
</tr>
<tr>
<td>00104A'</td>
<td>00104A'</td>
<td>HLL G,POINT 2</td>
<td></td>
</tr>
<tr>
<td>00104B'</td>
<td>00104B'</td>
<td>ISET LEFT HALF OF POINTER</td>
<td></td>
</tr>
<tr>
<td>00104C'</td>
<td>00104C'</td>
<td>JIRST I2</td>
<td></td>
</tr>
<tr>
<td>00104D'</td>
<td>00104D'</td>
<td>IIST CHARACTER IS ALREADY IN</td>
<td></td>
</tr>
<tr>
<td>00104E'</td>
<td>00104E'</td>
<td>NXTCHI</td>
<td></td>
</tr>
<tr>
<td>00104F'</td>
<td>00104F'</td>
<td>TTECALL 4,A</td>
<td></td>
</tr>
<tr>
<td>001050'</td>
<td>001050'</td>
<td>IINPUT CHARACTER</td>
<td></td>
</tr>
<tr>
<td>001051'</td>
<td>001051'</td>
<td>CAIGE A.,33</td>
<td></td>
</tr>
</tbody>
</table>
INSTRU  |  ADDRESS  |  OPERANDS  |  COMMENT
-------|-----------|------------|---------------------
0010  | 001460h  | 054 04 00 00 000001  | INSCRIPT TCALL 4,A  INPUT NEXT CHARACTER IN COMMAND STRING
0012  | 001461h  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0014  | 001462h  | 254 00 00 00 00015  | JOST -12  YES, GO TYPE MESSAGE
0016  | 001463h  | 306 31 00 00 00015  | CAIN A,19  END OF COMMAND?
0018  | 001464h  | 254 00 00 00 002242  | JOST ILLCHD  YES, GO TYPE MESSAGE
001A  | 001465h  | 265 17 00 00 001321  | JUMP B,MERR  IF NO, IS ZERO, GO TYPE MESSAGE
001C  | 001466h  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
001E  | 001467h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0020  | 001468h  | 051 04 00 00 003001  | TCALL 4,A
0022  | 001469h  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0024  | 00146Ah  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0026  | 00146Bh  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0028  | 00146Ch  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
002A  | 00146Dh  | 265 17 00 00 003713  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
002C  | 00146 Eh  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
002E  | 001470h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0030  | 001471h  | 051 04 00 00 003001  | TCALL 4,A
0032  | 001472h  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0034  | 001473h  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0036  | 001474h  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0038  | 001475h  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
003A  | 001476h  | 265 17 00 00 003713  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
003C  | 001477h  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
003E  | 001478h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0040  | 001479h  | 051 04 00 00 003001  | TCALL 4,A
0042  | 00147Ah  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0044  | 00147Bh  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0046  | 00147Ch  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0048  | 00147Dh  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
004A  | 00147 Eh  | 265 17 00 00 001371  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
004C  | 00147 Fh  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
004E  | 001480h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0050  | 001481h  | 051 04 00 00 003001  | TCALL 4,A
0052  | 001482h  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0054  | 001483h  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0056  | 001484h  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0058  | 001485h  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
005A  | 001486h  | 265 17 00 00 001371  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
005C  | 001487h  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
005E  | 001488h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0060  | 001489h  | 051 04 00 00 003001  | TCALL 4,A
0062  | 00148Ah  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0064  | 00148 Bh  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0066  | 00148Ch  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0068  | 00148 Dh  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
006A  | 00148 Eh  | 265 17 00 00 001371  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
006C  | 00148 Fh  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
006E  | 001490h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
0070  | 001491h  | 051 04 00 00 003001  | TCALL 4,A
0072  | 001492h  | 306 31 00 00 004040  | CAIN A,40  IS CHARACTER A SPACE?
0074  | 001493h  | 254 00 00 00 001471  | JOST -12  YES, GO INPUT NEXT CHARACTER
0076  | 001494h  | 302 31 00 00 000954  | CAIN A,15  IS CHARACTER A COMMA?
0078  | 001495h  | 254 00 00 00 001510  | JOST ISONE  YES, GO TO ISONE
007A  | 001496h  | 265 17 00 00 001371  | JUMP A,FSPRM  STORE FRAME NO, IN FSPRM
007C  | 001497h  | 322 02 00 00 003403  | MOVE B,FSPRM  STORE FRAME NO, IN FSPRM
007E  | 001498h  | 254 00 00 00 001472  | JOST +12  IGO CHECK 1ST CHARACTER AFTER NO.
SECTION H. EXECUTE "R" COMMAND. "R" COMMANDS CAUSE SELECTED LINES, IFRAMES, OR STRINGS TO BE REPLACED.

REPLCI TDCALL 4,A INPUT NEXT CHARACTER IN COMMAND STRING
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIN A,19 END OF COMMAND STRING?
Jrst ilcmd YES, GO TYPE ERROR MESSAGE (SECTION B)
Jsp G,1172 INPUT FRAME NO.
Jrst A,2 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIN A,19 END OF COMMAND STRING?
Jrst rlone YES, GO TO RPLONE
CAIE A, "I" IS CHARACTER A COMMA?
Jrst ilcmd NO, GO TYPE ERROR MESSAGE (SECTION B)
MOVEN B,FSTFRM ISTORE FRAME NO. IN FSTFRM
Jsp G,1172 ISTORE FRAME NO. IN FSTFRM
MOVEN B,FSTLIN ISTORE LINE NO. IN FSTLIN
Jrst A,40 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIE A, "I" IS CHARACTER A DASH?
Jrst roneln IND. GO TO RONELN
Jsp G,1172 IINPUT 2ND LINE NO.
Jrst A,2 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIE A,15 END OF COMMAND STRING?
Jrst ilcmd NO, GO TYPE ERROR MESSAGE (SECTION B)
MOVEN B,FLSTFRM ISTORE FRAME NO. IN FLSTFRM
Jsp G,1172 ISTORE FRAME NO. IN FLSTFRM
MOVEN B,FLSTLIN ISTORE LINE NO. IN FLSTLIN
Jrst A,40 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIE A,15 END OF COMMAND STRING?
Jrst ilcmd NO, GO TYPE ERROR MESSAGE (SECTION B)
MOVEN B,FLSTFRM ISTORE FRAME NO. IN FLSTFRM
Jsp G,1172 ISTORE FRAME NO. IN FLSTFRM
MOVEN B,FLSTLIN ISTORE LINE NO. IN FLSTLIN
Jrst A,40 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
Jrst A,2 YES, GO INPUT NEXT CHARACTER
CAIE A,15 END OF COMMAND STRING?
Jrst ilcmd NO, GO TYPE ERROR MESSAGE (SECTION B)
MOVEN B,FLSTFRM ISTORE FRAME NO. IN FLSTFRM
Jsp G,1172 ISTORE FRAME NO. IN FLSTFRM
MOVEN B,FLSTLIN ISTORE LINE NO. IN FLSTLIN
Jrst A,40 GO CHECK 1ST CHARACTER AFTER NO.
Jsp G,1172 TDCALL 4,A
CAIN A,40 IS CHARACTER A SPACE?
MOVEM B, FSTLN
ISAVE NO. IN FSTLN
JRST ADDR05

CAML CL, LSTFRM
IWILL CONSTANT BE ADDED TO LAST FRAME NO.
JRST +6
YES, SKIP 5 LINES

MOVEM B, FSTFRM
MOVE FRAME NO. TO FSTFRM

USERT 1,1
INPUT 1

JRT TTFNO
I GO TYPE ERROR MESSAGE (SECTION P)

JRT TTFNO
I WILL LAST FRAME NO. BE GREATER THAN 999?

JRT ADD06
IND. GO ADD CONSTANTS TO FRAME NO5.

TICALL 11,0

TICALL 3; CASCIE 

K COMMAND WOULD RESULT IN FRAME NUMBER GREATER THAN 999

/3

USERT 1,1
INPUT 1

JRT GETCMD
I GO ACCEPT NEXT COMMAND (SECTION B)

TICALL 11,0

TICALL 3; CASCIE

K COMMAND WOULD RESULT IN FRAME NUMBER LESS THAN OR EQUAL TO ZERO OR FRAME NUMBERS OUT OF SEQUENCE

/1

USERT 1,1
INPUT 1

JRT GETCMD
I GO ACCEPT NEXT COMMAND (SECTION B)

TICALL 11,0

JSP Q, INCHR
I INPUT CHARACTER

JSP Q, OUTCHR
I OUTPUT CHARACTER

JRT ADD06
YES, GO TO ADD06

JRT ADD07
BEGINNING OF FRAME?

JRT ADD08
YES, GO TO ADD08

JRT ADD09
END OF LESSON?

JRT ADD0A
YES, GO TO ADD0A

JRT ADD0B
IND. GO INPUT NEXT CHARACTER

JRT ADD0C
INPUT FRAME NO.

JRT ADD0D
ADD CONSTANT TO THIS FRAME?

JRT ADD0E
IND. GO OUTPUT FRAME NO.

JRT ADD0F
YES, ADD CONSTANT

JSP Q, IOO
I OUTPUT FRAME NO.

JSP Q, IOO
I OUTPUT SPACE AFTER NO.

JRT ADD07
I GO INPUT NEXT CHARACTER

CLOSE 1
I CLOSE OLD LESSON FILE

CLOSE 2
I CLOSE NEW LESSON FILE (SUPERCEDE OLD FILE)

JRT ADD03
CLOSE 1

CLOSE 2

SETM INAH=3

MOVIE B, "UNP"

MOVE B, ONAH=2

SETM ONAH=2

SETB 000180

LOOKUP 1, INAH
I LOOKUP NEW 'OLD' LESSON

JRT TTFQ2
I ON ERROR GO TYPE ERROR MESSAGE (SECTION D)
MOVE G,POINT2  ;PUT BYTE POINTER TO SLOT ZERO IN AC-G
IDB G       ;SKIP THE <56> AT BEGINNING OF LINE

1566       ;PUT NEW FRAME NO, IN 1ST LINE
1567       ;WRIT D, POLST-1 ;PUT PUSH DOWN POINTER IN AC-G
1568       ;DIVI A, 12         ;GET A DIGIT IN AC-B
1569       ;ADDI B, 6F         ;CONVERT IT TO AN ASCII CHARACTER
1570       ;PUSH D, B           ;PLACE CHARACTER IN PUSH DOWN LIST
1571       ;JUMP N, -3         ;POP D, A
1572       ;POP D, A
1573       ;IDPB A, G
1574       ;IDPB A, G
1575       ;JUMP C, -3
1576       ;IREWRITE REST OF 1ST LINE INTO BUFFER
1577      ;MOVEI A, 40
1578      ;IDPB A, G
1579      ;MOVEI A, "t"
1580      ;IDPB A, G
1581      ;MOVEI A, "y"
1582      ;IDPB A, G
1583      ;MOVEI A, "p"
1584      ;IDPB A, G
1585      ;MOVEI A, "n"
1586      ;IDPB A, G
1587      ;MOVEI A, "m"
1588      ;IDPB A, G
1589      ;MOVEI A, "s"
1590      ;IDPB A, G
1591      ;MOVE A, FMTYP
1592      ;IDPB A, G
1593      ;MOVE A, 15
1594      ;IDPB A, G
1595      ;MOVE A, 12
1596      ;IDPB A, G
1597      ;JSTR GETCHO    ;IGO ACCEPT NEXT COMMAND (SECTION B)
1598      ;EDLESI TRO FLAGS,1000 ;SET LAST FRAME FLAG
1599      ;SETM NTRFM ;SET NTRFM TO ZERO
1600      ;JSTR PUTINZ

SECTION K. EXECUTE AN "E", "ET", "EP", OR CARRIAGE RETURN
COMMAND. THESE COMMANDS TERMINATE LESSON EDITING.

1602      ;QUIT TRO FLAGS, 480 ;SET QUIT FLAG
1603      ;JSTR END2
1604      ;ENDI TRO FLAGS, 483 ;CLEAR QUIT, PROCESS, AND TEST FLAGS
1605      ;INPUT NEXT CHARACTER IN COMMAND STRING
1606      ;JSTR, 2 ;YES, GO INPUT NEXT CHARACTER
1607      ;CAI A, 40 ;IS CHARACTER A SPACE?
1608      ;CAI A, "T" ;IS CHARACTER A "T"?
1609      ;JSTR, 3 ;INO, SKIP 2 LINES
1610      ;TRO FLAGS, 2 ;YES, SET TEST FLAG

1611      ;TRO FLAGS, 0 ;NO, SET TEST FLAG
ISECTION L: INITIALIZE THE SPACE LIST BY PLACING THE NUMBERS
THROUGH 63 IN LOCATIONS SPACE THROUGH SPACE+63. THE SPACE
LIST CONTAINS THE NUMBERS OF THE EMPTY SLOTS IN THE EDITING
BUFFER.

ISECTION M: FILL THE EDITING BUFFER WITH THE NEXT FRAME OF THE
OLD LESSON FILE. EACH LINE IN THE FRAME IS PLACED IN A
SEPARATE SLOT IN THE EDITING BUFFER. THE NUMBER OF THE FRAME
BEING PLACED IN THE BUFFER IS STORED IN FRNUM. THE NUMBER OF
THE FOLLOWING FRAME IS ALSO READ IN AND PLACED IN NXTFRM, IF
THE LAST FRAME IS BEING PLACED IN THE BUFFER, NXTFRM IS SET TO
ZERO AND THE LAST FRAME FLAG IS SET. THE NUMBERS OF THE SLOTS
IN THE EDITING BUFFER WHICH HAVE BEEN FILLED ARE REMOVED FROM
THE SPACE LIST.

FILL: MOVEM G,PC2 ;SAVE RETURN ADDRESS

SETBP J,SPACE+1 ;INITIALIZE SPACE LIST (SECTION L)

SETBP I,8

POPE J,E ;REMOVE SLOT ZERO FROM SPACE LIST

MOVEI E,1

MOVEM E,LINK ;NEXT LINE WILL BE PLACED IN SLOT ONE

MOPE G,POINT2

MOPE G,BUF+1 ;HAS FRAME NUMBER BEEN READ?

JRT FINPT ;NO, GO READ IT

MOPE A,36 ;PLACE <36> AT BEGINNING OF FIRST LINE IN SLOT ZERO

MOPE A,NXTFRM ;GET CURRENT FRAME NO. FROM NXTFRM

MOPE A,FNUM ;PLACE FRAME NUMBER IN FNUM

MOPE A,NXTFRM ;PLACE FRAME NUMBER IN SLOT ZERO OF EDITING BUFFER

HRREI D,PDIST-5 ;PUT PUSH DOWN POINTER IN AC-D

IDIVI A,12 ;GET A DIGIT IN AC-B

IDIVI B,68 ;CONVERT IT TO AN ASCII CHARACTER

PUSHE B,D ;PLACE CHARACTER IN PUSH DOWN LIST

JUMP A,-3 ;GET CHARACTER FROM PUSH DOWN LIST

IDPB A,G ;PLACE CHARACTER IN THE EDITING BUFFER

IDPB A,G ;GET NO. OF CHARACTERS REMAINING IN AC-G

JUMP C,-3 ;PLACE SPACE AFTER FRAME NUMBER IN SLOT ZERO

MOVEI A,10 ;GO TO GET LPROS HIGH SEGMENT
1779        I NEW LESSON FILE, NO CHANGES ARE MADE IN THE SPACE LIST.
1776        I LINK LIST, FRNUM, NXTFRM, OR THE LAST FRAME FLAG.
1777        I
1778        002717                   MOVE O,PC2
1772        321 06 0 0 002717       ISAVE RETURN ADDRESS
1780        32721                   SET AF
1780        32721                   1579 LINE IS ALWAYS IN SLOT ZERO
1780        32721                   NXTL1 JUMP L,FPC2 IIF ALL LINES ARE OUT, RETURN
1781        32721                   MOV A,INLK (F) IGET SLOT NO. OF NEXT LINE IN AC-A
1782        32721                   MOVE G,F IPUT SLOT NO. OF CURRENT LINE IN AC-G
1783        32721                   MOVE F,A ISAVE SLOT NO. OF NEXT LINE IN AC-F
1784        I
1785        002725                   SET UP BYTE POINTER TO PROPER SLOT IN AC-G
1786        32725                   IMUL G,20 ICH LINE OCCUPIES 20 WORDS OF CORE
1787        32726                   271 07 0 0 002670c ADDI G,BUFF IADD ON BASE ADDRESS OF BUFFER
1788        002727                   500 00 0 0 002821c MUL G,POINT2 IGET LEFT HALF OF POINTER
1789        I
1790        002730                   IOUTPUT LINE INTO NEW LESSON FILE
1791        32730                   ILDB A,G IGET NEXT CHARACTER FROM BUFFER
1792        002731                   265 17 0 0 002575c JSP G,OUTCHR IOUTPUT CHARACTER
1793        002732                   336 01 0 0 002012c CAIN G,12 IEND OF LINE?
1794        002733                   32731                   JNSR NTXT IYES, GO SET UP BYTE POINTER TO NEXT LINE
1795        002734                   254 00 0 0 002790c JNST NTXT INO, GO OUTPUT NEXT CHARACTER
1796        I
1797        I
1798        ISECTION 0: TRANSFER REMAINING CHARACTERS IN THE 'OLD' LESSON FILE
1799        INTO THE 'NEW' LESSON FILE; CLOSE BOTH FILES SO THAT THE OLD LESSON
1800        IS SUPERCEDED, LOOKUP THE LESSON FILE, AND ENTER A 'NEW' LESSON FILE.
1801        IFRNUM AND NXTFRM ARE SET TO ZERO, AND THE LAST FRAME FLAG IS CLEARED.
1802        I
1803        002735                   282 17 0 0 003002c BEGI MOVE O,PC3 ISAVE RETURN ADDRESS
1804        002735                   336 00 0 0 002668c SKIPN FRNUM IALREADY AT BEGINNING?
1805        002735                   254 01 0 0 002735c JNST PC3 IYES, RETURN
1806        002740                   627 00 0 0 000001c JSTN FLAGS,1 ITEST AND CLEAR NO EMPTY FLAG
1807        002741                   265 17 0 0 002717c JSP G,EMPTY IEMPTY BUFFER IF NO EMPTY FLAG WAS NOT SET (SECTION N)
1808        002741                   602 00 0 0 001002c TRN FLAGS,1008 I1ST LINE IN BUFFER?
1809        002741                   254 01 0 0 000001c JNST LSTFRM IYES, GO TO LSTFRM
1810        002741                   32741                   201 01 0 0 000036c MOVEI A,36
1811        002745                   265 17 0 0 002731c JSP G,OUTCHR IOUTPUT <36>
1812        002745                   260 01 0 0 002715c MOVEA NXTFRM INEXT FRAME NO.
1813        002745                   265 17 0 0 002542c JSP C,100 JOUTPUT NEXT FRAME NO.
1814        002745                   265 17 0 0 002745c JSP C,OUTCHR IOUTPUT SPACE AFTER FRAME NO.
1815        002750                   201 01 0 0 000048c MOVEI A,40
1816        002751                   265 17 0 0 002745c JSP G,OUTCHR IREAD IN REST OF 'OLD' LESSON AND OUTPUT INTO 'NEW' LESSON FILE
1817        002752                   265 17 0 0 002672c OUT1 IJSP G,INCHR IINPUT CHARACTER
1818        002752                   265 17 0 0 002751c JSP G,OUTCHR IOUTPUT CHARACTER
1819        002754                   32754                   302 01 0 0 000004c CAIN G,4 IEND OF LESSON?
1820        002754                   254 02 0 0 002752c JNST OUT1 IYES, GO INPUT NEXT CHARACTER
1821        I
1822        002756                   32756                   073 02 0 0 002000c LLLL1 CLOSE 1 ICLOSE OLD LESSON FILE
1823        002757                   070 02 0 0 000000c LLLL2 CLOSE 2 ICLOSE NEW LESSON FILE (SUPERCEDES OLD LESSON)
1824        I
1825        ILOOKUP NEW 'OLD' LESSON AND ENTER 'NEW' LESSON
SECTION P. SEARCH FOR THE FRAME WITH THE FRAME NUMBER CONTAINED IN FRNFRM, AND PLACE IT IN THE EDITING BUFFER. IF THE FRAME IS NOT FOUND, PRINT AN ERROR MESSAGE AND GO TO ACCEPT THE NEXT EDIT COMMAND.

1858 0030F3  202    0    0    0    0    000000  MOVE Q:QC4  ISAVE RETURN ADDRESS
1859 0030F4  202    0    0    0    002365  MOVE B:FRNFRM  IF SEARCH IS FOR FRAME ZERO
1860 0030F5  322    0    0    0    003511  JUMPE B:TNTFND  ALREADY SEARCHED THIS FRAME NUMBER
1861 0030F6  316    0    0    0    003252  CAMN B:FRNFRM  IS DESIRED FRAME IN EDITING BUFFER?
1862 0030F7  254    0    0    0    003031  JST QC4  YES, RETURN
1863 0030F8  317    0    0    0    003066  CAMG B:FRNFRM  ALREADY PAST DESIRED FRAME?
1864 0030F9  265    0    0    0    002739  JSP Q:BEG  YES, GO TO BEGINNING OF LESSON (SECTION Q)
1865 0030FA  302    0    0    0    001090  TRNE FLGS,1000  ISA LAST FRAME IN BUFFER?
1866 0030FB  254    0    0    0    003047  JST FND2  YES, GO TO FND2
1867 0030FC  332    0    0    0    003010  SKIP FNFRM  IS THERE A FRAME IN THE EDITING BUFFER?
1868 0030FD  265    0    0    0    002717  JSP Q:EMPTY  YES, EMPTY BUFFER
1869 0030FE  332    0    0    0    002016  SKIP FRNFRM  WAS THERE A FRAME IN THE BUFFER?
1870 003100  254    0    0    0    003053  JST FNDFM3  YES, GO TO FNDFM3
1871 003101  265    0    0    0    002752  FNDFM21  INPUT <36>
1872 003102  265    0    0    0    002710  FNDFM2  INPUT FRAME NO.
1873 003103  202    0    0    0    002773  MOVE B:NXTFRM  IPUT FRAME NO. IN NXTFRM
1874 003104  200    0    0    0    003044  FNDFM31  MOVE B:FRNFRM  IIS THIS THE DESIRED FRAME?
1875 003105  316    0    0    0    003022  CAMN B:NXTFRM  YES, GO PUT IT IN THE BUFFER
1876 003106  254    0    0    0    003043  JST FMILL  YES, GO PUT IT IN THE BUFFER
1877 003107  201    0    0    0    003036  MOVEI A,36  JSP Q:OUTCHR  OUTPUT <36>.
1878 003108  265    0    0    0    003011  JSP Q:OUTCHR  OUTPUT FRAME NO.
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>003032'</td>
<td>MOVEI A, 48</td>
<td>MOVE I, FRAME NUMBER</td>
</tr>
<tr>
<td>003033'</td>
<td>JPNJ Q.OUTCHR</td>
<td>JUMP TO OUTPUT SPACE AFTER FRAME NO.</td>
</tr>
<tr>
<td>003034'</td>
<td>FNDFM41</td>
<td>JUMP TO FRAME NUMBER</td>
</tr>
<tr>
<td>003035'</td>
<td>JPNJ Q.INCHR</td>
<td>JUMP TO INPUT CHARACTER</td>
</tr>
<tr>
<td>003036'</td>
<td>FAIR A, 4</td>
<td>END OF LESSON</td>
</tr>
<tr>
<td>003037'</td>
<td>JRN FEND</td>
<td>YES, RETURN</td>
</tr>
<tr>
<td>003038'</td>
<td>JPNJ FEND</td>
<td>JUMP TO END OF LESSON</td>
</tr>
<tr>
<td>003039'</td>
<td>JRN FDNM2</td>
<td>YES, RETURN</td>
</tr>
<tr>
<td>003040'</td>
<td>JPNJ Q.OUTCHR</td>
<td>JUMP TO OUTPUT CHARACTER</td>
</tr>
<tr>
<td>003041'</td>
<td>JRN FDNM4</td>
<td>JUMP TO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>003042'</td>
<td>FILLI Q.FILL</td>
<td>PUT DESIRED FRAME IN BUFFER</td>
</tr>
<tr>
<td>003043'</td>
<td>FENDI TLO FLAGS.1</td>
<td>SET NO EMPTY FLAG</td>
</tr>
<tr>
<td>003044'</td>
<td>FDNM FEND2I</td>
<td>MAKE SURE FMNUM DOESN'T CONTAIN ZERO</td>
</tr>
<tr>
<td>003045'</td>
<td>TNCND JSP Q.BEG</td>
<td>GO TO BEGINNING OF LESSON</td>
</tr>
<tr>
<td>003046'</td>
<td>NTNCND TNCALL</td>
<td>CALL TO ASCIZ</td>
</tr>
<tr>
<td>003047'</td>
<td>TPNCND</td>
<td>FRAME IS NOT FOUND</td>
</tr>
<tr>
<td>003048'</td>
<td>TNCND JSP Q.IOT</td>
<td>TYPE NUMBER OF FRAME NOT FOUND</td>
</tr>
<tr>
<td>003049'</td>
<td>NTNCND TNCALL</td>
<td>ASCIZ NOT FOUND</td>
</tr>
<tr>
<td>00304A'</td>
<td>DN FEND</td>
<td>YES, RETURN</td>
</tr>
<tr>
<td>00304B'</td>
<td>JRN FDNM4</td>
<td>JUMP TO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>00304C'</td>
<td>FENDI TLO FLAGS.1</td>
<td>SET NO EMPTY FLAG</td>
</tr>
<tr>
<td>00304D'</td>
<td>FDNM FEND2I</td>
<td>MAKE SURE FMNUM DOESN'T CONTAIN ZERO</td>
</tr>
<tr>
<td>00304E'</td>
<td>TNCND JSP Q.BEG</td>
<td>GO TO BEGINNING OF LESSON</td>
</tr>
<tr>
<td>00304F'</td>
<td>NTNCND TNCALL</td>
<td>CALL TO ASCIZ</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED
PROGRAM BREAK IS 003436

4K CORE USED
TITLE AUX2

COMMENT

WRITTEN BY GARRET A. VANDER LUGT  COMPLETED 1-AUG-73

PART OF LEDIT.SHR, VERSION 3

AUX2.MAC CONTAINS A NUMBER OF SUBROUTINES WHICH ARE USED BY
PROGRAMS IN THE LEDIT HIGH SEGMENT,

CONTENTS OF AUX2.MAC

SECTION A, INPUTS A CHARACTER FROM A TERMINAL UNTIL THE END OF
A LINE IS REACHED.

SECTION B, INPUTS A CHARACTER FROM THE DISK.

SECTION C, OUTPUTS A CHARACTER TO THE DISK.

SECTION D, OUTPUTS A STRING TO THE DISK.

SECTION E, OUTPUTS AN INTEGER TO THE DISK.

SECTION F, OUTPUTS AN OCTAL NUMBER TO THE DISK.

SECTION G, OUTPUTS AN INTEGER TO A USER'S TERMINAL.

SECTION H, INPUTS AN INTEGER FROM A TERMINAL.

SECTION I, INPUTS AN INTEGER FROM THE DISK.

ENTRY POINTS TO AUX2.MAC

EDLIN = FROM LEDIT.MAC AND WRTEO.MAC, LOCATED IN SECTION A

IIO = FROM WRTEO.MAC, LOCATED IN SECTION I

IIIT = FROM PERL.MAC AND WRTEO.MAC, LOCATED IN SECTION H

IIITZ = FROM WRTEO.MAC, LOCATED IN SECTION H

INCHR = FROM PERL.MAC, WRTEO.MAC, AND AUX2.MAC, LOCATED IN
SECTION B

IOD = FROM PERL.MAC AND WRTEO.MAC, LOCATED IN SECTION E

IOD8 = FROM PERL.MAC, LOCATED IN SECTION F

IOT = FROM PERL.MAC AND WRTEO.MAC, LOCATED IN SECTION G
OUTCHR - FROM PRL.MAC, WRLED.MAC, AND AUX2.MAC, LOCATED IN SECTION C
OUTSTR - FROM PRL.MAC, LOCATED IN SECTION D

EXTERNAL MEMORY REFERENCES

IBUF = BUFFER HEADER FOR DISK INPUT (3 WORDS)
DBUF = BUFFER HEADER FOR DISK OUTPUT (3 WORDS)
PC2 = RETURN ADDRESS
PDLST = PUSH DOWN LIST FOR STORING ASCII DIGITS OF A NUMBER
BEING CONVERTED FROM BINARY TO ASCII CODE (8 WORDS)

ENTRY INCHR,OUTCHR,OUTSTR,IBUF,IBUF,PC2,PDLST

SECTION A, THIS SUBROUTINE INPUTS CHARACTERS FROM THE USER'S TERMINAL UNTIL AN END OF LINE IS ENCOUNTERED.

SECTION B, THIS SUBROUTINE INPUTS A CHARACTER FROM THE DISK
ION CHANNEL 1, THE CHARACTER IS PLACED IN AC-A.
AUX2 MACRO 47(204)-7 08134 2-AUG-74 PAGE 1-2
AUX2 MAC 6-JUN-74

107 02013 254 00 0 17 000000 JST (G) YES, RETURN
108 TTCALL 3: CASC2 /
109
110 00004 051 03 0 00 000124 /] DISK INPUT ERROR ON CHANNEL 1 = AUX2
111 00004 254 04 0 00 000000
112
113
114
115
116 ISECTION C. THIS SUBROUTINE OUTPUTS A CHARACTER TO THE DISK ON CHANNEL 2. THE CHARACTER MUST BE IN AC-A.
117
118 00004 375 00 0 00 000002
119 OUTFCHR SORGE OBUF+2 IDECREMENT CHARACTER COUNT
120 00004 051 03 0 00 000022
121 JST PUTBUF IF BUFFER FULL, GO EMPTY IT
122 00004 254 00 0 00 000001
123 PUTXII IDA A, OBUF+1 IPUT CHARACTER IN OUTPUT BUFFER
124 00004 254 00 0 17 000000 JST (G) RETURN
125 00004 257 00 0 00 000000
126 00004 254 00 0 00 000016 JST OUTFCHR IF NO ERROR, GO PUT CHARACTER IN BUFFER
127 TTCALL 3: CASC2 /
128
129 00004 051 03 0 00 000135 /] DISK OUTPUT ERROR ON CHANNEL 2 = AUX2
130 00004 254 04 0 00 000000
131
132
133
134 ISECTION D. THIS SUBROUTINE OUTPUTS A STRING OF CHARACTERS TO THE DISK. THE CHARACTERS MUST BE STORED IN ASCII FORMAT IMMEDIATELY FOLLOWING THE CALLING INSTRUCTION.
135
136 00004 050 02 0 00 000000 OUTSTR: MOVE B Q
137 00004 505 02 0 00 000000 HRL B.448700 SET UP POINTER IN AC-B
138 00004 134 00 0 00 000002 ILDB A, B GET CHARACTER IN AC-A
139 00004 322 01 0 02 000000 JUMPE A, 1 (B) RETURN ON NULL CHARACTER
140 00004 255 17 0 00 000016 JSN Q, OUTFCHR OUTPUT CHARACTER TO DISK
141 00004 254 00 0 00 000030 JST OUTSTR+2 IGO GET NEXT CHARACTER
142
143
144 ISECTION E. THIS SUBROUTINE OUTPUTS AN INTEGER TO THE DISK. THE INTEGER MUST BE IN AC-A.
145
146 00004 222 17 0 00 000000 ID1 MOVE Q, PC2 JSAVE RETURN ADDRESS
147 00004 551 04 0 00 00077777 HRZI D, POLST=1 SET UP PUSH DOWN POINTER IN AC-D
148 00004 231 01 0 00 000012 IDIVI A, 12 GET DIGIT IN AC-A
149 00004 435 02 0 00 000020 IDRI B, 63 JCONVERT TO ASCII CODE
150 00004 261 00 0 00 000002 PUSH D, B IPUT CHARACTER IN PUSH DOWN LIST
151 00004 326 01 0 00 000036 JUMP A, 100+2 IGO GET NEXT DIGIT IF THERE IS ONE
152 00004 262 00 0 00 000016 JOP D, B GET CHARACTER FROM PUSH DOWN LIST
153 00004 265 17 0 00 000016 JSN Q, OUTFCHR OUTPUT CHARACTER TO DISK
154 00004 683 04 0 00 00077777 TNE D, 077777 ANY CHARACTERS LEFT?
155 00004 254 00 0 00 000042 JST =-3 YES, GO GET NEXT CHARACTER
156 00004 254 02 1 00 000034 JST QPC2 OTHERWISE, RETURN
157
158
159
SECTION F. THIS SUBROUTINE OUTPUTS AN OCTAL NUMBER TO THE
IDISK. THE NUMBER MUST BE IN AC-A.

160 20067' 202 17 0 00 000046' 10081 MOVE Q,PC2 1SAVE RETURN ADDRESS
165 20067' 551 04 0 00 77777' HRRZI D,PDST=1 1SET UP PUSH DOWN POINTER IN AC-D
167 20067' 231 01 0 00 00010' IDIVI A,10 1GET DIGIT IN AC-A
169 20067' 255 02 0 00 00000' IOR1 A:0 1GET INTEGER CODE
170 20067' 261 04 0 00 00002' PUSH D,B 1PUT CHARACTER IN PUSH DOWN LIST
171 20067' 20067' 262 04 0 00 00003' JUMPN A,10006+2 1GO GET NEXT DIGIT IF THERE IS ONE
172 20067' 265 17 0 00 00316' JSR G.OUTCHR 1OUTPUT CHARACTER TO DISK.
173 20067' 603 02 0 00 77777' TLANE D,77777 1ANY CHARACTERS LEFT?
174 20067' 254 00 0 00 00005' JRST =-3 1YES, GO GET NEXT CHARACTER
175 20067' 254 00 1 00 00004' JRST OPC2 1OTHERWISE, RETURN

SECTION G. THIS SUBROUTINE OUTPUTS AN INTEGER TO THE USER'S
TERMINAL. THE INTEGER MUST BE IN AC-A.

180 20067' 202 17 0 00 000061' I0T1 MOVE Q,PC2 1SAVE RETURN ADDRESS
185 20067' 551 04 0 00 77777' HRRZI D,PDST=1 1SET UP PUSH DOWN POINTER IN AC-D
187 20067' 231 01 0 00 00012' IDIVI A,12 1GET DIGIT IN AC-A
189 20067' 255 02 0 00 00000' IOR1 A:0 1GET INTEGER CODE
190 20067' 261 04 0 00 00002' PUSH D,B 1PUT CHARACTER IN PUSH DOWN LIST
191 20067' 20067' 256 04 0 00 00004' JUMPN A,10006+2 1GO GET NEXT DIGIT IF THERE IS ONE
192 20067' 262 04 0 00 00001' POP D,A 1GET CHARACTER FROM PUSH DOWN LIST
193 20067' 20067' 051 01 0 00 00001' TCALL 1,1/A 1TYPE THE CHARACTER
194 20067' 254 00 0 00 77777' TLANE D,77777 1ANY CHARACTERS LEFT?
195 20067' 254 00 1 00 00070' JRST =-3 1YES, GO GET NEXT CHARACTER
196 20067' 254 00 1 00 00062' JRST OPC2 1OTHERWISE, RETURN

SECTION H. THIS SUBROUTINE INPUTS AN INTEGER FROM A USER'S
TERMINAL AND PLACES IT IN AC-B. IF THE I1T2 ENTRY POINT IS
USED, THEN THE FIRST DIGIT MUST ALREADY BE IN AC-A. IF THE
I1T1 ENTRY POINT IS USED, LEADING SPACES ARE IGNORED.

199 200075' 400 02 0 00 00000' I1T1 SETE B,
200 200076' 051 04 0 00 00001' TCALL 4,4/A 1INPUT ASCII CHARACTER
202 200077' 336 01 0 00 00000' CAIN A:40 1IS CHARACTER A SPACE?
203 200077' 254 00 0 00 00000' JRST =-2 1YES, GO INPUT NEXT CHARACTER
204 200077' 254 00 1 00 00010' JRST +5 1
205 200077' 200076' 051 04 0 00 00001' I1T1 TCALL 4,4/A 1INPUT NEXT CHARACTER
206 200077' 330 01 0 00 00002' I1T3 ICALL A:60
207 200077' 330 01 0 00 02071' I1T31 CAIL A:71
208 200077' 254 30 0 00 00000' JRTC (Q) 1RETURN IF CHARACTER IN NOT A DIGIT
209 200077' 200061' 221 02 0 00 00012' IMULP BY 10
210 200077' 200076' 271 02 0 00 77772 1ADD VALUE OF DIGIT TO NO. IN AC-B
211 200077' 254 00 0 00 000102' JRST I1T1 1GO GET NEXT DIGIT
212 200077' 400 02 0 00 00000' I1T21 SETE B,
SECTION 1. THIS SUBROUTINE INPUTS AN INTEGER FROM THE DISK AND
PLACES IT IN AC-B.

213 000112' 254 00 00 000031' JAST '113
214
215
216
217
218
219 000113' 262 17 00 000016' ID1
220 000114' 400 02 00 000000' MOVEB Q,16
221 000115' 265 17 00 0000041' ID1
222 000116' 301 01 00 000060' JSP Q,INCHR
223 000117' 303 01 00 000071' CAIL A,60
224
225 000120' 254 00 00 000080' ID1
226 000121' 221 02 00 000002' JRST (16)
227 000122' 271 02 01 777720' ADDI B,-60(A)
228 000123' 254 00 00 000105' JRST '113
229
230
231

NO ERRORS DETECTED

PROGRAM BREAK IS 000146

3K CORE USED
TITLE LDTLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 1-AUG-73

PART OF LEDIT,SHR, VERSION 3

LDTLOW,MAC IS A DUMMY LOW SEGMENT WHICH IS USED TO LABEL MEMORY
LOCATIONS IN THE LOW SEGMENT SO THAT THEY CAN BE EASILY REFERENCED
BY PROGRAMS IN THE LEDIT HIGH SEGMENT, SINCE NO INSTRUCTIONS OR
CONSTANTS ARE STORED IN THE LOW SEGMENT, NO BINARY CORE IMAGE OF IT
IS PRODUCED BY THE SSAVE MONITOR COMMAND.

ENTRY ARGMT

INTERN NXTFRM,FRMINC,INBUF,BUFFER,LINBUF,OLDSTR,TRMBLK,JOBNO,
INTERN FRMNUM,FRTYPE,LINK,SPACE,FSTFRM,LSTFRM,LSTLIN,LSTLIN,
INTERN STORE,INAH,ONAH,PC1,INBUF,OBUF,PC2,POSLIN,PPN,PG3,CRTNUM,PG4
INTERN INBUF,OUTBP2,OUTBUF

00137  JOBSN
00313  LOC JOBSN
00301  AND 381,1
00000  RELCO 0

00000  STORE; BLOCK 1 IUSED BY LEDIT,MAC, PERL,MAC, LPRINT,MAC, AND WRTED,MAC
00001  ARGMT; BLOCK 6 IUSED BY LEDIT,MAC AND WRTED,MAC
00002  JORN; BLOCK 1 IJOB NUMBER
00003  TRMBLK; BLOCK 3 ITRMBP, ARGUMENTS
00004  INTBLK; BLOCK 5 I INTERRUPT BLOCK IS STORED HERE
00005  INLOC; BLOCK 12 I INTERRUPT ROUTINE IS STORED HERE
00006  IBUF; BLOCK 3 IUSED BY WRTED,MAC AND AUX2,MAC
00007  OBUF; BLOCK 3 IUSED BY WRTED,MAC AND AUX2,MAC
00008  INAH; BLOCK 4 IUSED BY PERL,MAC AND WRTED,MAC
00009  ONAH; BLOCK 4 IUSED BY PERL,MAC, LPRINT,MAC, AND WRTED,MAC
00010  PPNI; BLOCK 1 IUSED BY PERL,MAC
00011  INBUF; BLOCK 283 IUSED BY PERL,MAC AND WRTED,MAC
00012  OUTBUF; BLOCK 283 IUSED BY WRTED,MAC

JANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN LDTLOW,MAC,
IRPTLOW,MAC, AND FRSLow,MAC.

00457  PC11; BLOCK 1 IUSED BY PERL,MAC AND WRTED,MAC
00460  PC21; BLOCK 1 IUSED BY WRTED,MAC AND AUX2,MAC
00461  PC31; BLOCK 1 IUSED BY WRTED,MAC
00462  PC41; BLOCK 1 IUSED BY WRTED,MAC
00463  POLSL; BLOCK 10 IUSED BY WRTED,MAC AND AUX2,MAC
00473  FRMINC; BLOCK 1 IUSED BY WRTED,MAC
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Description</th>
<th>Used By</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>003474'</td>
<td>EDTNUM: BLOCK 1</td>
<td>PERL, MAC</td>
</tr>
<tr>
<td>55</td>
<td>003475'</td>
<td>FSTLIN: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>56</td>
<td>003476'</td>
<td>LSTLIN: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>57</td>
<td>003479'</td>
<td>FSTRHM: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>58</td>
<td>003509'</td>
<td>LSTRHM: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>59</td>
<td>003591'</td>
<td>NXTFRHM: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>60</td>
<td>002582'</td>
<td>FRMNUM: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>61</td>
<td>002583'</td>
<td>FRMTYP: BLOCK 1</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>62</td>
<td>007584'</td>
<td>OLOSTRI: BLOCK 3</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>63</td>
<td>002587'</td>
<td>LINBUF: BLOCK 20</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>64</td>
<td>000527'</td>
<td>LINK: BLOCK 100</td>
<td>LPRINT, MAC AND</td>
</tr>
<tr>
<td>65</td>
<td>000627'</td>
<td>SPACEI: BLOCK 100</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>66</td>
<td>000727'</td>
<td>INBUF2: BLOCK 200</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>67</td>
<td>001132'</td>
<td>OUTBF2: BLOCK 200</td>
<td>WRTED, MAC</td>
</tr>
<tr>
<td>68</td>
<td>001365'</td>
<td>BUFFER: BLOCK 1</td>
<td>WRTED, MAC (1ST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>WORD OF 4K EDT</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 001336

3K CORE USED
TITLE LRPRPT

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT COMPLETED 16-JUL-73

PART OF LRPRPT.SHR, VERSION 3

SPECIAL NOTE

LRPRPT.MAC CONTAINS THE ONLY ENTRY POINT TO THE LRPRPT HIGH SEGMENT.
THIS ENTRY POINT IS LOCATED IN SECTION B OF THIS PROGRAM.

THE LRPRPT HIGH SEGMENT IS COMPOSED OF THE FOLLOWING SUBPROGRAMS:

1. LRPRPT.MAC
2. RPRT.MAC
3. LRPRPT.MAC
4. RPRT.MAC
5. AUXS.MAC
6. RPTLOW.MAC (DUMMY LOW SEGMENT)

THESE PROGRAMS SHOULD BE LOADED IN THE ORDER IN WHICH THEY ARE LISTED
ABOVE, AFTER LOADING, A REENTRANT, BINARY IMAGE OF THE LRPRPT
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SAVE MONITOR COMMAND.

THE LRPRPT HIGH SEGMENT EXECUTES THE FOLLOWING LEARN SYSTEM COMMANDS:

1. BYE
2. CLEAR
3. DELETE
4. KJOB
5. MONITOR
6. REPORT
7. UPDATE

LRPRPT.MAC INTERPRETS LEARN SYSTEM COMMANDS AND CALLS THE PROPER SUB-
PROGRAMS TO EXECUTE THEM. IF A COMMAND CANNOT BE EXECUTED BY THE
LRPRPT HIGH SEGMENT, THEN LRPRPT.MAC PLACES A COMMAND INDEX IN LOCATION
STORE AND REPLACES ITSELF WITH THE HIGH SEGMENT WHICH CAN EXECUTE THE
COMMAND. WHEN LRPRPT.MAC IS ENTERED FROM ANOTHER HIGH SEGMENT, IT
RECOVERS A COMMAND INDEX FROM LOCATION STORE AND CALLS THE PROPER
SUBPROGRAM TO EXECUTE THE COMMAND INDICATED BY THE COMMAND INDEX.

CONTENTS OF LRPRPT.MAC
SECTION A. PERFORMS THE GETSEG UDO.

SECTION B. CONTAINS THE ENTRY POINT TO THE LRPRTR HIGH SEGMENT.

SECTION C. INPUTS THE FIRST WORD IN A LEARN COMMAND.

SECTION D. INTERPRETS THE FIRST WORD IN A LEARN COMMAND.

SECTION E. EXECUTES A "MONITOR" COMMAND.

SECTION F. SETS UP GETSEG ARGUMENTS FOR THE LEARN HIGH SEGMENT.

SECTION G. SETS UP GETSEG ARGUMENTS FOR THE LEDIT HIGH SEGMENT.

SECTION H. SETS UP GETSEG ARGUMENTS FOR THE LPROS HIGH SEGMENT.

SECTION I. CALLS THE PROPER LRPRTR SUBPROGRAM TO EXECUTE A LEARN COMMAND.

SECTION J. EXECUTES A "BYE" COMMAND.

SECTION K. EXECUTES A "KJOB" COMMAND.

ENTRY POINT TO LRPRTR.MAC

RETURN - FROM ANOTHER HIGH SEGMENT, LOCATED IN SECTION B

EXTERNAL SUBROUTINES CALLED BY LRPRTR.MAC

CLR - EXECUTES A "CLEAR" COMMAND, LOCATED IN RPRT.MAC (SECTION D)

DEL - EXECUTES A "DELETE" COMMAND, LOCATED IN RPRT.MAC (SECTION A)

EOLIN - INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE END OF A LINE IS REACHED, LOCATED IN AUX3.MAC (SECTION A)

RPRT - EXECUTES A "REPORT" COMMAND, LOCATED IN RPRT.MAC (SECTION A)

UPDATE - EXECUTES AN "UPDATE" COMMAND, LOCATED IN RPRT.MAC (SECTION E)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN RPDTLOW.MAC)

ARGINT - ARGUMENTS FOR GETSEG UDO (6 WORDS)

STORE - COMMAND INDEX IS SAVED HERE DURING A GETSEG

TRMBLK - TRHOP, ARGUMENTS (3 WORDS)
LRPRT MACRO 47(204)-7 06138 2-Aug-74 PAGE 1-2

LRPRT MACRO 6-JUN-74

107
108
109
110 000001
111
112 EXTERN ARGXT, CLR, DEL, EOLN, RPT, STORE, UPDT, TRNBLK
113
114 00001 A#1
115 00002 B#2
116 00003 C#3
117 00004 D#4
118 00005 E#5
119 00006 F#6
120 00007 G#7
121 00008 Q#17
122
123
124 ISECTIO A. GET THE PROPER HIGH SEGMENT TO EXECUTE THE COMMAND
125 WHICH WAS TYPED BY THE USER. EXECUTION RESUMES IN LEARN, MAC,
126 LEDIT, MAC OR LPROG, MAC.
127
128 NOTE: NO DATA OR INSTRUCTIONS SHOULD BE PLACED AHEAD OF
129 "GETSG", SINCE THE RETURN ADDRESS OF THE GETSEG UTO MUST BE
130 THE SAME IN ALL HIGH SEGMENTS.
131
132 000008 047 01 0 00 000040 GETSG: GETSEG A.
133 000008 047 01 0 00 000040 GETSG: GETSEG A.
134
135
136 ISECTIO B. RETRIEVE THE COMMAND INDEX WHICH WAS PLACED IN LOCATION
137 "STORE" BY THE PREVIOUS HIGH SEGMENT AND JUMP TO THE PROPER ROUTINE
138 TO EXECUTE THE COMMAND.
139
140 007002 200 02 0 00 000000* RETURN MVEN B, STORE IGET THE COMMAND INDEX IN AC-B
141 002003 254 00 1 02 000031 JST R.(B)
142 002004 00000 000020 NV 0, REPORT ISEE SECTION 1
143 002005 00000 000011 NV 0, UPDATE ISEE SECTION 1
144 007006 00000 000021 NV 0, CLEAR ISEE SECTION 1
145 002007 00000 000027 NV 0, DELETE ISEE SECTION 1
146
147
148 ISECTIO C. INPUT THE FIRST WORD IN A LEARN SYSTEM COMMAND.
149
150 002010 000 00 000000* CMDI MVEN A, TRNBLK IMOVE ADDRESS OF TRMOP, ARGUMENTS TO AC-A
151 00011 047 01 0 00 00116 TRMOP, A, ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY
152 002012 254 00 0 00 00116 JST EMPTY
153 002013 201 00 0 00 00001 MVEN 0, 1
154 000014 047 02 0 00 00031 SLEEP 8, I SLEEP A SECS BEFORE CHECKING BUFFER AGAIN
155 000015 254 00 0 00 00111 JST , -4 I GO CHECK BUFFER AGAIN
156 000016 251 14 0 00 00000 EMPTY 1, TTDCALL 14 I OVERRIDE USER'S CONTROL 0
157 000017 255 00 0 00 00000 JFCL
158 TTDCALL 3, EASCIE /
159
160 ENTER COMMAND
SECTION J. EXECUTE THE BYE COMMAND BY RUNNING THE LOGOUT PROGRAM.

323  002215'  63 71  63  00  00  00
324  002216'  54  57  47  57  65  64
325  002217'  200  09  00  00  000000
326  002218'  000  09  00  00  000000
327  002219'  000  00  00  00  000000
328  002220'  000  00  00  00  000000
329  002221'  000  00  00  00  000000
330  002222'  051  11  00  000000
331  002223'  047  01  00  000035
332  002224'  254  04  00  000000

BYE1

TTCALL 11:0

MOVE A, RUNARG1

INPUT ADDRESS OF RUN ARGUMENTS IN AC-A

RUN A,

RUN THE LOGOUT SYSTEM PROGRAM

HALT

SECTION K. EXECUTE THE KJOB COMMAND BY RUNNING THE KJOB PROGRAM.

338  002227'  63  71  63  00  00  00
339  002230'  53  52  57  42  00  00
340  002231'  000  00  00  000000
341  002232'  000  00  00  000000
342  002233'  000  00  00  000000
343  002234'  000  00  00  000000
344  002235'  254  00  00  000237'
345  002236'  254  00  00  000236'
346  002237'  305  01  00  000040
347  002238'  254  32  00  000036'
348  002239'  305  01  00  000057
349  002240'  254  02  00  000226'
350  002241'  254  03  00  000259'
351  002242'  254  03  00  000015
352  002243'  051  04  00  000001
353  002244'  201  01  00  000227'
354  002245'  047  01  00  000035
355  002246'  254  04  00  000000
356  002247'  047  01  00  000035
357  002248'  254  04  00  000000

RARG: SIXBIT /SYS/

KJOB: JRST +2

TTCALL 4:A

INPUT NEXT CHARACTER IN COMMAND STRING

CAIN A: 4E

IS CHARACTER A SPACE?

JRSA: "2"

YES, GO INPUT NEXT CHARACTER

CAIN A:"/

IS CHARACTER A SLASH?

JRS KJ

YES, GO TO KJ

CAIN A: 15

END OF COMMAND STRING?

JRSA ERR

NO, GO TYPE ERROR MESSAGE (SECTION G)

TTCALL 4:A

INPUT LINE FEED

MOVE A: RARG

INPUT ADDRESS OF RUN ARGUMENTS IN AC-A

RUN A,

RUN THE KJOB SYSTEM PROGRAM

HALT

NOSYM

END

NO ERRORS DETECTED

PROGRAM BREAK IS 008308

3K CORE USED
RPRT
MACRO 47(204)-7 00138 2-AUG-74 PAGE 1

TITLE RPRT

COMMENT %

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 11-MAR-74

PART OF LRPT, SHR, VERSION 3

RPRT,MAC EXECUTES DELETE, CLEAR, AND UPDATE COMMANDS. IT ALSO
GENERATES LESSON REPORTS (SUMMARY FORM), STUDENT REPORTS (SUMMARY
FORM), AND ANSWER LISTINGS. LRPT,MAC GENERATES DETAILED REPORTS.

CONTENTS OF RPRT,MAC

SECTION A. INTERPRETS REPORT AND DELETE COMMAND STRINGS.

SECTION B. PICKS UNIQUE NAME FOR REPORT FILES.

SECTION C. INITIALIZES THE REPORT GENERATOR.

SECTION D. GENERATES LESSON REPORTS (SUMMARY FORM).

SECTION E. GENERATES STUDENT REPORTS (SUMMARY FORM).

SECTION F. GENERATES ANSWER LISTINGS.

SECTION G. EXECUTES DELETE COMMANDS.

SECTION H. EXECUTES CLEAR COMMANDS.

SECTION I. EXECUTES UPDATE COMMANDS.

LEARN RECORD FILE FORMATS

1. TRANS,REC - DATA IS WRITTEN INTO TRANS,REC BY INTERP,MAC AT
   THE COMPLETION OF LESSON EXECUTION (BINARY DATA). THE
   LESSON DATA IS STORED IN THE FOLLOWING FORMAT:

   1ST WORD = NO. OF WORDS STORED FOR THIS LESSON
   2ND WORD = LESSON NAME (SIXBIT)
   3RD WORD = STUDENT PROJ.-PROC NO.
   4TH WORD = DATE OF EXECUTION
   5TH WORD = STARTING TIME FOR LESSON (MSECS)
   6TH WORD = ELAPSED LESSON TIME (MSECS)
   7TH WORD = NO. CORRECT
   8TH WORD = NO. INCORRECT
   9TH WORD = NO. NEUTRAL
  10TH WORD = NO. OF TIME UPS
   11TH WORD = TOTAL ANSWERING TIME (SECS)
   12TH WORD = TOTAL NO. OF FRAMES
13TH WORD = CPU TIME (MSECS)

IN ADDITION, TWO WORDS OF DATA ARE STORED FOR EACH FRAME
EXECUTED. THE DATA FORMAT FOR O-FRAMES IS AS FOLLOWS:

1ST WORD = BITS 0-101 FRAME NO.
BITS 11-171 LETTER INDICATING FRAME TYPE
BITS 18-351 DAYTIME IN SECONDS

2ND WORD = BITS 0-51 ANSWER TAG (SEVEN BIT CHARACTER)
BITS 5-121 RIGHT-WRONG-NEUTRAL INDICATOR
BITS 12-231 ANSWERING TIME IN SECONDS
BIT 25 IS SET IF CALC MODE WAS USED

FOR OTHER FRAME TYPES, THE 1ST WORD IS THE SAME AS FOR
O-FRAMES, AND THE 2ND WORD IS ZERO.

2. MASTER.REC = DATA IS APPENDED ONTO MASTER.REC EACH TIME
AN UPDATE COMMAND IS ISSUED, DATA FORMAT IS THE SAME AS FOR
TRANS.REC EXCEPT THAT THE 1ST WORD IN THE FILE CONTAINS THE
DATE OF THE UPDATE AND THE 2ND WORD CONTAINS THE TIME OF
THE UPDATE.

3. ANSWER FILES - ONE LINE OF ASCII INFORMATION IS WRITTEN INTO
THE ANSWER FILE FOR EACH ANSWER TYPED IN BY A STUDENT. THE
DATA FORMAT FOR EACH LINE IS AS FOLLOWS:
CHARACTERS 1-41 FRAME NO.
CHARACTERS 5-101 STUDENT PROG. NO.
CHARACTER 111 ANSWER TAG
CHARACTER 121 RIGHT-WRONG-NEUTRAL INDICATOR
REST OF LINE = STUDENT ANSWER
NOTE: EACH LINE IS TERMINATED BY \(<15>ONLY.

ENTRY POINTS TO RPRT.MAC
CLR = FROM LRPT.MAC (SECTION I), LOCATED IN SECTION H
DEL = FROM LRPT.MAC (SECTION I), LOCATED IN SECTION A
RPRT = FROM LRPT.MAC (SECTION I), LOCATED IN SECTION A
UPDT = FROM LRPT.MAC (SECTION I), LOCATED IN SECTION I

EXTERNAL SUBROUTINES CALLED BY RPRT.MAC
II = INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B;
LOCATED IN AUX3.MAC (SECTION C)
II12 = INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>1ST DIGIT IS ALREADY IN AC-A, LOCATED IN AUX3.MAC (SECTION C)</td>
</tr>
<tr>
<td>108</td>
<td></td>
</tr>
<tr>
<td>109</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>INCHR - INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN AUX3.MAC (SECTION B)</td>
</tr>
<tr>
<td>111</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td>INWDR - INPUTS A WORD OF DATA FROM THE DISK INTO AC-A, LOCATED IN AUX3.MAC (SECTION G)</td>
</tr>
<tr>
<td>114</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>IOC - OUTPUTS THE INTEGER IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION D)</td>
</tr>
<tr>
<td>117</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td></td>
</tr>
<tr>
<td>119</td>
<td>LPRRT - GENERATES DETAILED REPORTS, LOCATED IN LPRRT.MAC (SECTION A)</td>
</tr>
<tr>
<td>120</td>
<td></td>
</tr>
<tr>
<td>121</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td>OUTFAR2 - OUTPUTS THE DATE IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION J)</td>
</tr>
<tr>
<td>123</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td>OUTFAR2 - OUTPUTS THE TIME IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION F)</td>
</tr>
<tr>
<td>126</td>
<td></td>
</tr>
<tr>
<td>127</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>OUTCHR - OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION K)</td>
</tr>
<tr>
<td>129</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td></td>
</tr>
<tr>
<td>131</td>
<td>OUTDAT - OUTPUTS THE CURRENT DATE TO THE DISK, LOCATED IN AUX3.MAC (SECTION J)</td>
</tr>
<tr>
<td>132</td>
<td></td>
</tr>
<tr>
<td>133</td>
<td></td>
</tr>
<tr>
<td>134</td>
<td>OUTLES - OUTPUTS THE LESSON NAME IN AC-D TO THE DISK, LOCATED IN AUX3.MAC (SECTION I)</td>
</tr>
<tr>
<td>135</td>
<td></td>
</tr>
<tr>
<td>136</td>
<td></td>
</tr>
<tr>
<td>137</td>
<td>OUTPPN - OUTPUTS THE PROGRAMMER NO. IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION L)</td>
</tr>
<tr>
<td>138</td>
<td></td>
</tr>
<tr>
<td>139</td>
<td></td>
</tr>
<tr>
<td>140</td>
<td>OUTSTR - OUTPUTS A STRING OF CHARACTERS TO THE DISK, LOCATED IN AUX3.MAC (SECTION E)</td>
</tr>
<tr>
<td>141</td>
<td></td>
</tr>
<tr>
<td>142</td>
<td></td>
</tr>
<tr>
<td>143</td>
<td>OUTTIM - OUTPUTS THE CURRENT TIME TO THE DISK, LOCATED IN AUX3.MAC (SECTION K)</td>
</tr>
<tr>
<td>144</td>
<td></td>
</tr>
<tr>
<td>145</td>
<td></td>
</tr>
<tr>
<td>146</td>
<td>OUTTM - OUTPUTS LESSON TIME OR ANSWERING TIME FROM AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION M)</td>
</tr>
<tr>
<td>147</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>OUTWDR - OUTPUTS A WORD OF DATA FROM AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION W)</td>
</tr>
<tr>
<td>150</td>
<td></td>
</tr>
<tr>
<td>151</td>
<td></td>
</tr>
<tr>
<td>152</td>
<td>RTJUST - RIGHT JUSTIFIES THE NUMBER IN AC-A, LOCATED IN AUX3.MAC (SECTION N)</td>
</tr>
<tr>
<td>153</td>
<td></td>
</tr>
<tr>
<td>154</td>
<td></td>
</tr>
<tr>
<td>155</td>
<td>RPRINT - PLACES THE REPORT FILE IN PRINT QUEUE, LOCATED IN RPRINT.MAC (SECTION A)</td>
</tr>
<tr>
<td>156</td>
<td></td>
</tr>
<tr>
<td>157</td>
<td></td>
</tr>
<tr>
<td>158</td>
<td></td>
</tr>
<tr>
<td>159</td>
<td>EXTERNAL MEMORY REFERENCES (ALL IN RTPLOW.MAC)</td>
</tr>
</tbody>
</table>

**Note:** The text appears to be a list of macro definitions or functions, possibly from a software development context.
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>ABO = REPORT COMMAND SWITCH</td>
</tr>
<tr>
<td>161</td>
<td></td>
</tr>
<tr>
<td>162</td>
<td></td>
</tr>
<tr>
<td>163</td>
<td>ANSTOT = TOTAL ANSWERING TIME IS ACCUMULATED HERE</td>
</tr>
<tr>
<td>164</td>
<td></td>
</tr>
<tr>
<td>165</td>
<td>DATA = LIST OF SUMMARY DATA (16 WORDS)</td>
</tr>
<tr>
<td>166</td>
<td></td>
</tr>
<tr>
<td>167</td>
<td>DAYT = DATES SPECIFIED IN REPORT COMMAND (2 WORDS)</td>
</tr>
<tr>
<td>168</td>
<td></td>
</tr>
<tr>
<td>169</td>
<td>DEV = DEVICE SPECIFIED IN REPORT COMMAND</td>
</tr>
<tr>
<td>170</td>
<td></td>
</tr>
<tr>
<td>171</td>
<td>EXTOT = NO. OF TIMES LESSON WAS EXECUTED</td>
</tr>
<tr>
<td>172</td>
<td></td>
</tr>
<tr>
<td>173</td>
<td>FRAME = FRAME NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)</td>
</tr>
<tr>
<td>174</td>
<td></td>
</tr>
<tr>
<td>175</td>
<td>IBUF = DISK BUFFER HEADER (3 WORDS)</td>
</tr>
<tr>
<td>176</td>
<td></td>
</tr>
<tr>
<td>177</td>
<td>IBUFF = DISK INPUT BUFFER (131 WORDS)</td>
</tr>
<tr>
<td>178</td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>IBUF2 = DISK BUFFER HEADER (3 WORDS)</td>
</tr>
<tr>
<td>180</td>
<td></td>
</tr>
<tr>
<td>181</td>
<td>INAM = LOOKUP ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>182</td>
<td></td>
</tr>
<tr>
<td>183</td>
<td>INAM2 = LOOKUP ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>184</td>
<td></td>
</tr>
<tr>
<td>185</td>
<td>LESSON = LESSON NAME SPECIFIED IN REPORT COMMAND</td>
</tr>
<tr>
<td>186</td>
<td></td>
</tr>
<tr>
<td>187</td>
<td>LIM1 = USED TO STORE INDEX BY QUICKSORT</td>
</tr>
<tr>
<td>188</td>
<td></td>
</tr>
<tr>
<td>189</td>
<td>LIM2 = USED TO STORE INDEX BY QUICKSORT</td>
</tr>
<tr>
<td>190</td>
<td></td>
</tr>
<tr>
<td>191</td>
<td>MONTH = MONTH SPECIFIED IN REPORT OR DELETE COMMAND</td>
</tr>
<tr>
<td>192</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>NTTOT = NO. OF NEUTRAL RESPONSES IS ACCUMULATED HERE</td>
</tr>
<tr>
<td>194</td>
<td></td>
</tr>
<tr>
<td>195</td>
<td>OBUF = DISK BUFFER HEADER (3 WORDS)</td>
</tr>
<tr>
<td>196</td>
<td></td>
</tr>
<tr>
<td>197</td>
<td>OBUFF = DISK OUTPUT BUFFER (131 WORDS)</td>
</tr>
<tr>
<td>198</td>
<td></td>
</tr>
<tr>
<td>199</td>
<td>OBUF2 = DISK BUFFER HEADER (3 WORDS)</td>
</tr>
<tr>
<td>200</td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>ONAM = ENTER ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>202</td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>ONAM2 = ENTER ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>204</td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>PARAM = BEGINNING OF FRAME DATA LIST</td>
</tr>
<tr>
<td>206</td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>RC1 = RETURN ADDRESS IS STORED HERE</td>
</tr>
<tr>
<td>208</td>
<td></td>
</tr>
<tr>
<td>209</td>
<td>PPNI = PROGRAMMER NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)</td>
</tr>
<tr>
<td>210</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td>RESTOP = TOTAL NO. OF RESPONSES IS ACCUMULATED HERE</td>
</tr>
<tr>
<td>212</td>
<td></td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION A. INTERPRET REPORT AND DELETE COMMAND STRINGS.

DEL1 TROD FLAGS, 10 ; SET THE DELETE COMMAND FLAG, AND SKIP
RPR1 TRZ FLAGS, 10 ; CLEAR THE DELETE COMMAND FLAG
MOVE Q, PC1 ; SAVE THE RETURN ADDRESS
SEZM LESSON ; CLEAR THE LESSON NAME
SEZM DEV ; CLEAR THE DEVICE NAME
SEZM PPNL1 ; CLEAR THE 1ST PROGRAMMER NO.
SEZM PPNL2 ; CLEAR THE 2ND PROGRAMMER NO.
SEZM DALT ; CLEAR THE 1ST DATE
SEZM DALT+1 ; CLEAR THE 2ND DATE
SEZM DALT+2 ; CLEAR THE 3RD DATE
SEZM FRAME+1 ; CLEAR THE 1ST FRAME NO.
SEZM FRAME+2 ; CLEAR THE 2ND FRAME NO.
SEZM FRAME+3 ; CLEAR THE 3RD FRAME NO.
SEZM DABO ; CLEAR THE COMMAND SWITCH
MOVE B, pointer1 ; GET THE BYTE POINTER
TTCALL A, A ; INPUT NEXT CHARACTER IN COMMAND STRING
CAlN A, 48 ; IS CHARACTER A SPACE?
CAI IN A, 12 ; YES, G0 INPUT NEXT CHARACTER
CAI N A, 15
RJST CMeRR ; IF END OF COMMAND, GO TYPE ERROR MESSAGE
TRMN FLG, 10 ; IS THIS A REPORT COMMAND?
RJST, 93 ; YES, SKIP
RJST GETDAT ; YES, G0 INPUT DATE
CAI N A, "1" ; CHARACTtoi A "1"?
CAlN A, "1" ; YES, G0 INPUT NEXT CHARACTER
RJST GETNUM ; IF CHARACTER IS NOT A LETTER, GO TO GETNUM
MOVE C, 6
RJST, 2
TTCALL A, A ; INPUT NEXT CHARACTER IN LESSON OR DEVICE NAME
CAI N A, 48
CAI N A, 2
CAI N A, "1"
RJST, 5 ; IF NOT A LETTER, JUMP
ADDI A, 48 ; CONVERT CHARACTER TO SIXBIT
IDPB A, B ; PLACE CHARACTER IN AC-0
SOUGE C, GNAH ; GO INPUT NEXT CHARACTER
RJST NAHERL ; IF MORE THAN 8 CHARACTERS, GO TYPE ERROR MESSAGE
SENDI A, 0 ;
SEZM DABO ;
TTCALL 11, 3 ; CLEAR TTY INPUT BUFFER
CAI N A, 71
RJST NAMIN ; IF CHARACTER IS NOT A NUMBER, GO TO NAMIN
ADDI A, 48
IDPB A, 0
SOUGE C, GNAH
NAHERL ;
TTCALL 3, ASCII1
CAI N A, 71
TOO MANY CHARACTERS IN LESSON NAME
CAI N A, 11
RETURN TO LRPR1, MAC
SECTION B. PICK A UNIQUE NAME FOR THE REPORT FILE.

TRYAGN MOVE C.POINT2 IGET BYTE POINTER
MOVEI B,L IMAKE 1ST LETTER OF NAME 'L'
IDPB B,C IADD IT TO THE FILENAME
IDIV A,12 IGET LAST DIGIT IN AC-8
ADDI B,20 ICONVERT IT TO A SIXBIT CHARACTER
IDPB B,C IADD IT TO THE FILENAME
IDIV A,12 IGET NEXT TO LAST DIGIT IN AC-8
ADDI B,20 ICONVERT IT TO A SIXBIT CHARACTER
IDPB B,C IADD IT TO THE FILENAME
MOVE B,THPI HRIRH.DONAM IADD 1THPI TO THE FILENAME
MOVS B,LLE1 ISET EXTENSION TO 'LST'
MOVEI B,ONAM-I ISET NAME IN USER'S DIRECTORY
JRST +2 IF NONE FOUND, JUMP
JRST TRYAGN ELSE, GO PICK ANOTHER NAME

SECTION C. INITIALIZE THE REPORT GENERATOR.

CLOSE 1;
SETSTS 1,13 ICHANGE TO BINARY MODE
MOVE B,44000000000I MOVE B,IBM1 ISET UP BUFFER BYTE POINTER FOR BINARY INPUT
MOVE B,NAM2 MOVEM B.INAM IPUT 'MASTER' IN NAM
MOVS B,REC1 MOVEM B.INAM+1 IPUT REC1 IN NAM+1
SETFM INAM+3 SETFM INAM3 ILOOKUP MASTER RECORD FILE
JRST +2 IF NOT FOUND, GO TYPE ERROR MESSAGE
SETFM ONAM+2
SETFM ONAM+3
ENTER 2,ONAM IENTER REPORT FILE
JRST NOPROG IIF NO PROGRAM ON DISK, GO TYPE ERROR MESSAGE
SETFM EXTOT ISET NO. OF LESSONS EXECUTED TO ZERO
SETFM TINTOT ICLEAR THE TOTAL ELAPSED TIME
SETFM NSTOT ICLEAR THE TOTAL ANSWERING TIME
SETFM RSTOT ICLEAR THE TOTAL NO. OF RESPONSES
SETFM WSTOT ICLEAR THE NO. OF CORRECT RESPONSES
SETFM NSTOT ICLEAR THE NO. OF INCORRECT RESPONSES
SETFM TSTOT ICLEAR THE NO. OF NEUTRAL RESPONSES
SETFM TUPSTOT ICLEAR THE NO. OF TIME UPS
SETFM RUNTOT ICLEAR THE TOTAL CPU TIME
MOVS C,LPT1 MOVS C,DEV I11ST DEVICE LPT1?
MOVS C,DEV IMove B,ABO IGET COMMAND SWITCH
<table>
<thead>
<tr>
<th>Line</th>
<th>Code 1</th>
<th>Code 2</th>
<th>Code 3</th>
<th>Code 4</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>611</td>
<td>002613</td>
<td>265 17</td>
<td>0 00</td>
<td>000650*</td>
<td>JSP Q,OUTSTR OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>612</td>
<td>002614</td>
<td>040 255</td>
<td>040 000 000</td>
<td>ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>613</td>
<td>002615</td>
<td>203 01</td>
<td>0 00</td>
<td>00001*</td>
<td>MOVE A, DAY+1 GET THE 2ND DATE</td>
</tr>
<tr>
<td>614</td>
<td>002616</td>
<td>265 17</td>
<td>0 00</td>
<td>000612*</td>
<td>JSP Q,OUTDATE OUTPUT THE 2ND DATE</td>
</tr>
<tr>
<td>615</td>
<td>002617</td>
<td>265 17</td>
<td>0 00</td>
<td>000613*</td>
<td>JSP Q,OUTSTR OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>616</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>617</td>
<td>002620</td>
<td>015 012</td>
<td>123 124 125</td>
<td>STUDENT /</td>
<td></td>
</tr>
<tr>
<td>618</td>
<td>002621</td>
<td>104 125 116 124 123</td>
<td>002405</td>
<td></td>
<td></td>
</tr>
<tr>
<td>619</td>
<td>002622</td>
<td>072 040 000 000 000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>620</td>
<td>002623</td>
<td>200 01</td>
<td>0 00</td>
<td>000240*</td>
<td>MOVE A, PPNL GET THE 1ST PROGRAMMER NO.</td>
</tr>
<tr>
<td>621</td>
<td>002624</td>
<td>265 17</td>
<td>0 00</td>
<td>000020*</td>
<td>JSP Q,OUTPPN OUTPUT THE NO.</td>
</tr>
<tr>
<td>622</td>
<td>002625</td>
<td>201 01</td>
<td>0 00</td>
<td>000055</td>
<td>MOVE A, &quot;\n&quot;</td>
</tr>
<tr>
<td>623</td>
<td>002626</td>
<td>265 17</td>
<td>0 00</td>
<td>000020*</td>
<td>JSP Q,OUTCHR OUTPUT A &quot;\n&quot;</td>
</tr>
<tr>
<td>624</td>
<td>002627</td>
<td>203 01</td>
<td>0 00</td>
<td>00001*</td>
<td>MOVE A, PPNL+1 GET THE 2ND PROGRAMMER NO.</td>
</tr>
<tr>
<td>625</td>
<td>002628</td>
<td>265 17</td>
<td>0 00</td>
<td>000240*</td>
<td>JSP Q,OUTPPN OUTPUT THE NO.</td>
</tr>
<tr>
<td>626</td>
<td>002629</td>
<td>622 00</td>
<td>0 00</td>
<td>000310</td>
<td>TRE FLAGS.T0 CLEAR THE STUDENT REPORT FLAG</td>
</tr>
<tr>
<td>627</td>
<td>002630</td>
<td>254 02</td>
<td>0 00</td>
<td>000705</td>
<td>JSTR FNDBEG GO TO FNDBEG</td>
</tr>
<tr>
<td>628</td>
<td>002631</td>
<td>662 00</td>
<td>0 00</td>
<td>000310</td>
<td>HEADR! TRNFLG .T0 IS THE STUDENT REPORT FLAG SET?</td>
</tr>
<tr>
<td>629</td>
<td>002632</td>
<td>265 17</td>
<td>0 00</td>
<td>000666*</td>
<td>JSTR HEADR YES, GO TO HEADR</td>
</tr>
<tr>
<td>630</td>
<td>002633</td>
<td>269 17</td>
<td>0 00</td>
<td>000617*</td>
<td>JSP Q,OUTSTR OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>631</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>632</td>
<td>002634</td>
<td>015 012</td>
<td>015 012</td>
<td>STUDENT DATE E,TIME A,TIME RIGHT WRONG NEUT TIME UP</td>
<td></td>
</tr>
<tr>
<td>633</td>
<td>002635</td>
<td>125 125</td>
<td>105 116</td>
<td></td>
<td></td>
</tr>
<tr>
<td>634</td>
<td>002636</td>
<td>011 043</td>
<td>040 040</td>
<td>0104</td>
<td></td>
</tr>
<tr>
<td>635</td>
<td>002637</td>
<td>101 124 105 040 040</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>636</td>
<td>002638</td>
<td>040 105 056 124 111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>637</td>
<td>002639</td>
<td>125 105 040 121</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>638</td>
<td>002640</td>
<td>256 124 111 115 109</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>639</td>
<td>002641</td>
<td>040 042 040 040 122</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>640</td>
<td>002642</td>
<td>111 127 110 124 400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>641</td>
<td>002643</td>
<td>040 127 122 117 116</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>642</td>
<td>002644</td>
<td>107 047 040 116 125</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>643</td>
<td>002645</td>
<td>125 124 211 112 111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>644</td>
<td>002646</td>
<td>115 125 040 125 120</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>645</td>
<td>002647</td>
<td>015 012 015 012 003</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>646</td>
<td>002648</td>
<td>254 00 0 00 00741*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>647</td>
<td>002649</td>
<td>265 17</td>
<td>0 00</td>
<td>000635*</td>
<td>JSP Q,OUTSTR OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>648</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>649</td>
<td>002650</td>
<td>015 012</td>
<td>015 012</td>
<td>STUDENT /</td>
<td></td>
</tr>
<tr>
<td>650</td>
<td>002651</td>
<td>125 125</td>
<td>105 116</td>
<td></td>
<td></td>
</tr>
<tr>
<td>651</td>
<td>002652</td>
<td>011 043</td>
<td>040 040</td>
<td>0104</td>
<td></td>
</tr>
<tr>
<td>652</td>
<td>002653</td>
<td>101 124 105 040 040</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>653</td>
<td>002654</td>
<td>040 105 056 124 111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>654</td>
<td>002655</td>
<td>125 105 040 121</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>655</td>
<td>002656</td>
<td>256 124 111 115 109</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>656</td>
<td>002657</td>
<td>040 042 040 040 122</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>657</td>
<td>002658</td>
<td>111 127 110 124 400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>658</td>
<td>002659</td>
<td>040 127 122 117 116</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>659</td>
<td>002660</td>
<td>107 047 040 116 125</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>660</td>
<td>002661</td>
<td>125 124 211 112 111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>661</td>
<td>002662</td>
<td>115 125 040 125 120</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Macro Description:**
- **JSP Q,OUTSTR:** Outputs the following string.
- **ASCIZ /:** Initial state.
- **MOVE A, DATE+1:** Moves the date to the next day.
- **JSP Q,OUTDATE:** Outputs the current date.
- **JSP Q,OUTSTR:** Outputs the following string.
- **ASCIZ /:** Initial state.
- **MOVE A, PPNL:** Moves to the next programmer.
- **JSP Q,OUTPPN:** Outputs the programmer no.
- **JOIN FNDBEG:** Moves to the next record.
- **HEARDT! TRNFLG:** Checks if the student report flag is set.
- **JSTR HEADR:** Moves to the next student.
- **JSTR OUTD:** Moves to the next student.
- **HEADR! TRNFLG:** Checks if the student report flag is set.

**Program Flow:**
1. Initial state.
2. Move to the next day.
3. Output the current date.
4. Move to the next programmer.
5. Output the programmer no.
6. Clear the student report flag.
7. Move to the next record.
8. Check if the student report flag is set.
9. Move to the next student.
10. Move to the next student.
11. Check if the student report flag is set.

**Sample Output:**
- Student Date: 12/12/00
- Time: 01:00
- Right: Yes
- Wrong: No
- Neut: Yes
- Time Up: 01:00
<table>
<thead>
<tr>
<th>RPT</th>
<th>MACRO</th>
<th>47(204)-7.08138 2-AUG-74 PAGE 1-15</th>
</tr>
</thead>
<tbody>
<tr>
<td>748</td>
<td>022753</td>
<td>204 01 0 000840</td>
</tr>
<tr>
<td>749</td>
<td>022754</td>
<td>265 17 0 002620</td>
</tr>
<tr>
<td>750</td>
<td>022755</td>
<td>203 01 0 000354</td>
</tr>
<tr>
<td>751</td>
<td>022756</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>752</td>
<td>022757</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>753</td>
<td>022760</td>
<td>201 01 0 003110</td>
</tr>
<tr>
<td>754</td>
<td>022761</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>755</td>
<td>022762</td>
<td>202 01 0 000302</td>
</tr>
<tr>
<td>756</td>
<td>022763</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>757</td>
<td>022764</td>
<td>265 17 0 002664</td>
</tr>
<tr>
<td>758</td>
<td>022765</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>759</td>
<td>022766</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>760</td>
<td>022767</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>761</td>
<td>022768</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>762</td>
<td>022769</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>763</td>
<td>022770</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>764</td>
<td>022771</td>
<td>265 17 0 000764</td>
</tr>
<tr>
<td>765</td>
<td>022772</td>
<td>265 17 0 000111</td>
</tr>
<tr>
<td>766</td>
<td>022773</td>
<td>265 17 0 000044</td>
</tr>
<tr>
<td>767</td>
<td>022774</td>
<td>243 04 0 000063</td>
</tr>
<tr>
<td>768</td>
<td>022775</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>769</td>
<td>022776</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>770</td>
<td>022777</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>771</td>
<td>022778</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>772</td>
<td>022779</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>773</td>
<td>022780</td>
<td>265 17 0 000764</td>
</tr>
<tr>
<td>774</td>
<td>022781</td>
<td>265 17 0 000111</td>
</tr>
<tr>
<td>775</td>
<td>022782</td>
<td>265 17 0 000044</td>
</tr>
<tr>
<td>776</td>
<td>022783</td>
<td>265 17 0 000063</td>
</tr>
<tr>
<td>777</td>
<td>022784</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>778</td>
<td>022785</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>779</td>
<td>022786</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>780</td>
<td>022787</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>781</td>
<td>022788</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>782</td>
<td>022789</td>
<td>265 17 0 000764</td>
</tr>
<tr>
<td>783</td>
<td>022790</td>
<td>265 17 0 000111</td>
</tr>
<tr>
<td>784</td>
<td>022791</td>
<td>265 17 0 000044</td>
</tr>
<tr>
<td>785</td>
<td>022792</td>
<td>265 17 0 000063</td>
</tr>
<tr>
<td>786</td>
<td>022793</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>787</td>
<td>022794</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>788</td>
<td>022795</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>789</td>
<td>022796</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>790</td>
<td>022797</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>791</td>
<td>022798</td>
<td>265 17 0 000764</td>
</tr>
<tr>
<td>792</td>
<td>022799</td>
<td>265 17 0 000111</td>
</tr>
<tr>
<td>793</td>
<td>022800</td>
<td>265 17 0 000044</td>
</tr>
<tr>
<td>794</td>
<td>022801</td>
<td>265 17 0 000063</td>
</tr>
<tr>
<td>795</td>
<td>022802</td>
<td>265 17 0 000000</td>
</tr>
<tr>
<td>796</td>
<td>022803</td>
<td>265 17 0 000663</td>
</tr>
<tr>
<td>797</td>
<td>022804</td>
<td>265 17 0 000756</td>
</tr>
<tr>
<td>798</td>
<td>022805</td>
<td>265 17 0 000302</td>
</tr>
<tr>
<td>799</td>
<td>022806</td>
<td>265 17 0 000616</td>
</tr>
<tr>
<td>800</td>
<td>022807</td>
<td>265 17 0 000000</td>
</tr>
</tbody>
</table>
ADD I TO THE NO. OF NEUTRAL RESPONSES FROM OTHER LESSONS
ADD I TO THE TOTAL RESPONSES FOR ALL LESSONS
JSP Q,GTJUST IRIGHT JUSTIFY NO.
JSP Q,10D OUTPUT THE NO. OF NEUTRAL RESPONSES
MOVEI A,46 IOUTPUT A SPACE
MOVEI A,40 IINPUT NO. BACK IN AC-A
MOVE AC-I INPUT NO. BACK IN AC-A
JSP Q,GTJUST IRIGHT JUSTIFY NO.
JSP Q,10D OUTPUT NO. OF TIME UPS
JSP Q,OUTSTR IOUTPUT THE FOLLOWING STRING
ASCII /
MOVE B,ABO IGET NO. OF DATA WORDS FOR THIS LESSON
SUBI B,14 ISUBTRACT NO. OF WORDS ALREADY READ
JSP Q,INCHR IINPUT DATA WORD
JSP Q,INCHDI INPUT DATA WORDS FOR THIS LESSON ARE IN;GO TO END
JST I-2 IGO INPUT NEXT DATA WORD
NOTRAN II CALL 3,ASCII /
MASTER FILE NOT FOUND
CLOSE 1;
CLOSE 2;
SETSTS 1,0 IRESTORE ASCII MODE
MOVE B,C7EOOO0000001
MOVEM B,1BUFF+1 IRESTORE ASCII BYTE POINTER
ENDFL I TRUE FLAGS,10 IIS THE A STUDENT REPORT?
JST ENDFL IYES, GO TO ENDFL
JSP Q,OUTSTR IOUTPUT THE FOLLOWING STRING
ASCII /
NO. OF TIMES LESSON WAS EXECUTED /
MOVE A,EXTOT IGET THE TOTAL NO. OF TIMES LESSON WAS EXECUTED
JSP Q,10D OUTPUT TOTAL NO. OF EXECUTIONS
JST QEPD ILEAF EXECUTED AT LEAST ONCE
JSP Q,OUTSTR IOUTPUT THE FOLLOWING STRING
ASCII /
JST RESET IGO TO RESET
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>847</td>
<td>800116</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>848</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>849</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>850</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>851</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>852</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>853</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>854</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>855</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>856</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>857</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>858</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>859</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>860</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>861</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>862</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>863</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>864</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>865</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>866</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>867</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>868</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>869</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>870</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>871</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>872</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>873</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>874</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>875</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>876</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>877</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>878</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>879</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
<tr>
<td>880</td>
<td>800117</td>
<td><code>265 17 0 00 001113</code></td>
</tr>
</tbody>
</table>
IDIVI A, *D18
CALL B, 5
AQS A
;ROUND OFF
IDIVI A, *D18
MOTION B,HOTOT
;SAVE FRACTIONAL PART IN HOTOT
JSP Q, 10D
;OUTPUT THE WHOLE NO. PART
MOVE B, "n"
JSP Q, OUTCHR
;OUTPUT A DECIMAL POINT
MOVE A, HOTOT
;GET THE FRACTIONAL PART
JSP Q, 10D
;OUTPUT FRACTIONAL PART
JSP Q, OUTSTR
OUTPUT THE FOLLOWING STRING
ASCII / %

RESET: CLOSE 2,
CLOSE 1,
SETSTS 1:0
RESTORE ASCII MODE
MOVE A, [700000000]
MOVE B, 1BUFF+1
RESTORE ASCII BYTE POINTER
MOVE B, "T"Y
CAM B, DEV
;IS OUTPUT DEVICE TTY?
JAST TYPFIL
;YES, GO TO TYPFIL
JSP Q, PRINT
ELSE, PLACE REPORT FILE IN PRINT QUEUE
TCALL 3, ASC2 / 5

REPORT FILE IN PRINT QUEUE
JAST QPC1
;RETURN TO LRPRRT, MAC
TYPFIL
MOVE B, ONAM
GET THE REPORT FILE NAME
MOVE B, INAM
PUT IT IN INAM
MOVE B, LIST+1
MOVE B, INAM+1
PUT LIST IN INAM+1
SETEM INAM+3
LOOKUP 1, INAM
LOOKUP REPORT FILE
JAST Q, INCHR
;INPUT CHARACTER FROM REPORT FILE
STATE 1, 20000
END OF FILE?
JAST +3
;YES, JUMP
TCALL 1, A
;ELSE TYPE CHARACTER
JAST +4
;GO INPUT NEXT CHARACTER
SETEM INAM
SETEM INAM+3
RENAME 1, INAM
DELETE REPORT FILE
HALT
JAST QPC1
;RETURN TO LRPRRT, MAC

SECTION E. PRODUCE A STUDENT REPORT (SUMMARY FORM).

STORPT: SETEM STOT1 SET NO. OF STUDENTS EXECUTING LESSONS TO ZERO
JSP Q, OUTSTR
OUTPUT THE FOLLOWING STRING
ASCII /
"LEARN STUDENT REPORT" ( SUMMARY FORM )
MOVE A, WGTOT
GET FRACTIONAL PART

JSP Q,IOD
OUTPUT FRACTIONAL PART

JSP Q,OUTSTR
OUTPUT THE FOLLOWING STRING

ASCIIZ /

SKPOUT: AOS PPNL INCREMENT PROGRAMMER NO.

MOVE A, PPNL
PUT PROG. NO. IN AC-A

CAMEL A, PPNL+1 HAVE ALL STUDENTS BEEN INCLUDED?

JRST ALLOUT: YES, GO TO ALLOUT

SETZ EXTOT: SET NO. OF LESSONS EXECUTED TO ZERO

SETZ ANSTOT: ICLEAR THE TOTAL ANSWERING TIME

SETZ ERRTO: ICLEAR NO. OF RESPONSES

SETZ RTTOT: ICLEAR NO. OF CORRECT RESPONSES

SETZ WGTOT: ICLEAR NO. OF WRONG RESPONSES

SETZ NTTOT: ICLEAR NO. OF NEUTRAL RESPONSES

SETZ TPTTOT: ICLEAR NO. OF TIME UPS

SETZ RUNTIM: ICLEAR THE TOTAL CPU TIME

SETZ TINTOT: ICLEAR THE TOTAL ELAPSED TIME

USEIT 1,1: RETURN TO BEGINNING OF MASTER FILE

JSP Q,INCHR: INPUT UPDATE TIME

JSP Q,INCHR: INPUT UPDATE DATE

JRST FNBEG: GO TO FNBEG (SECTION D)

ALLOUT: JSP Q,OUTSTR OUTPUT THE FOLLOWING STRING

NO. OF STUDENTS EXECUTING LESSONS /

MOVE A, STDTOT: GET TOTAL NO. OF STUDENTS WHO EXECUTED LESSONS

JSP Q,IOD: OUTPUT THE NO.

JSP Q,OUTSTR: OUTPUT THE FOLLOWING STRING

ASCIIZ /

JRST RESET: GO TO RESET (SECTION D)

SECTION F: PRODUCE AN ANSWER LISTING.

TRZ FLAGS: 2 I CLEAR TTY FLAG

HLF R: 1 DEE: 0 SUB: DEE SPECIFIED IN REPORT COMMAND

CARG B: TTY1: 155 DEVICE TTY?

TRG FLAGS: 2 IYES, SET TTY FLAG

SKIPB E W: ES DEVICE UNSPECIFIED?

TRG FLAGS: 2 IYES, SET TTY FLAG

SKIP E LESSON HAS A LESSON NAME SPECIFIED?

JRST QUE2: IYES, GO TO QUE2
MOVEL A,LIM1
MOVEL B,TMP1:TOP
PUSH 14, OSORT
MOV 15, TOP1
PUSH 14, OSORT
POP 15, B
ENDP

GOBK27
MLE A, PARAH (C) I GET FRAME NO. FROM RETRIEVAL LIST
JMP HE A, ALLFIN IF END OF LIST, GO TO ALLFIN
MLEH PARAH (G) I CLEAR RIGHT HALF OF PRECEDING ITEM
HRRE C, PARAH (G) I CHARACTER COUNT IN AC-C
CAH A, PARAH (G) I ITS NEW FRAME NO. SAME AS LAST FRAME NO.?
JST LONLINS IYES, GO TO LONLINS
ASP 15, 2
JSP G, OUTSTR ELSE, OUTPUT THE FOLLOWING STRING
NOLINS I TAKES FLAGS: 2 I IS LISTING TO LPT1?
JST 1,*4 INC, JUMP
MOVE A, 11 PUT TAB IN AC-A
JSP G, OUTCHR I OUTPUT TAB
JSP G, OUTCHR I OUTPUT TAB
IDIV C, 1200 I DIVIDE CHARACTER COUNT BY NO. OF CHARACTERS PER BLOCK
ADV C IADD ONE TO GET DISK BLOCK CONTAINING THE LINE
USRI 1, (C)
INPUT 1, IINPUT THAT BLOCK
MOVE A, IBUF 2 I GET NO. OF CHARACTERS IN THE BLOCK
SUB A, D I COMPUTE NO. OF CHARACTER AFTER BEGINNING OF THE LINE
HMOVE A, IBUF 2 I SET CHARACTER COUNT EQUAL TO THIS NO.
IDIV D, 5 I COMPUTE WORD COUNT FOR THE LINE
ADD D, 16 I ADD IT TO BUFFER BYTE POINTER
JUMPE E, + 3 IF POINTER IS AT BEGINNING OF LINE, JUMP
ISP BUF 2 I INCREMENT BUFFER BYTE POINTER
SOJA E, + 2 I IF STILL NOT AT BEGINNING OF LINE, INCREMENT AGAIN
MOVEL B, 4
JSP G, INCR I INPUT FRAME NO. DIGIT
JSP G, OUTCHR I OUTPUT FRAME NO. DIGIT
JSP G, OUTCHR I OUTPUT FRAME NO. DIGIT
JSP G, OUTCHR I INPUT NEXT DIGIT, IF ANY
MOVEL A, 15
JSP G, OUTCHR I OUTPUT A TAB
MOVEL B, 6
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
REPRODUCED WITH PERMISSION OF THE COPYRIGHT OWNER. FURTHER REPRODUCTION PROHIBITED WITHOUT PERMISSION.
```
RPT  MACRO  47(204)-7  28358  2-AUG-74  PAGE 1-38

1438  TCALL 3,CASCIE /
1439  ANSWER FILE DELETED
1440  JST  @PC1  IRETURN TO LRPRTP,MAC
1441  0022701 851 03 0 0033691 /
1442  NDANSF1: TCALL 3,CASCIE /
1443  ANSWER FILE NOT FOUND
1444  JST  @PC1  IRETURN TO LRPRTP,MAC
1445  DELLESI: MOVE B,LESON IGET THE SPECIFIED LESSON NAME
1446  MOVE B,INAH IPUT IT IN INAH
1447  MOVE C,ABD IGET THE SPECIFIED SWITCH
1448  CAIN C,"B" IWA S "B" SWITCH SPECIFIED?
1449  JST  @PC1  IYES, JUMP
1450  CAIN C,"U" IWAS "U" SWITCH SPECIFIED?
1451  JST  @PC1  IYES, JUMP
1452  JST  DELPRS IYES, GO TO DELPRS
1453  JST  DELPRS IYES, GO TO DELPRS
1454  JST  DELPRS IYES, GO TO DELPRS
1455  JST  DELPRS IYES, GO TO DELPRS
1456  JST  DELPRS IYES, GO TO DELPRS
1457  JST  DELPRS IYES, GO TO DELPRS
1458  JST  DELPRS IYES, GO TO DELPRS
1459  JST  DELPRS IYES, GO TO DELPRS
1460  JST  DELPRS IYES, GO TO DELPRS
1461  JST  DELPRS IYES, GO TO DELPRS
1462  JST  DELPRS IYES, GO TO DELPRS
1463  JST  DELPRS IYES, GO TO DELPRS
1464  JST  DELPRS IYES, GO TO DELPRS
1465  JST  DELPRS IYES, GO TO DELPRS
1466  TCALL 3,CASCIE /
1467  UNPROCESSED LESSON DELETED
1468  UNPFND1: TCALL 3,CASCIE /
1469  UNPROCESSED LESSON NOT FOUND
1470  DELPRS: MOVE B,LESON IGET THE SPECIFIED LESSON NAME
1471  MOVE B,INAH IPUT IT IN INAH
1472  MOVE B,INAH IPUT IT IN INAH
1473  MOVE B,INAH IPUT IT IN INAH
1474  MOVE C,ABD IGET THE SPECIFIED SWITCH
1475  CAIN C,"U" IWAS "U" SWITCH SPECIFIED?
1476  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1477  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1478  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1479  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1480  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1481  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1482  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1483  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1484  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1485  JST  @PC1  IYES, RETURN TO LRPRTP,MAC
1486  TCALL 3,CASCIE /
1487  PROCESSED LESSON DELETED
1488  TCALL 3,CASCIE /
1489  TCALL 3,CASCIE /
1490  TCALL 3,CASCIE /
```
...
section 1. execute update commands.

1649 003062' 202 17 0 0 0 03061' UPDATE MOVE G.PC1 SAVE RETURN ADDRESS
1651 003063' 051 04 0 0 0 00001' TDCALL 4,A INPUT CHARACTER FROM TERMINAL
<table>
<thead>
<tr>
<th>Field</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
<th>Value 4</th>
<th>Value 5</th>
<th>Value 6</th>
<th>Value 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRMARR</td>
<td>459</td>
<td>467</td>
<td>476#</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>253#</td>
<td>1109</td>
<td>1144</td>
<td>1145</td>
<td>1150</td>
<td>1159</td>
<td>1160</td>
</tr>
<tr>
<td></td>
<td>1299</td>
<td>1300</td>
<td>1301</td>
<td>1304</td>
<td>1649</td>
<td>1647</td>
<td>1649</td>
</tr>
<tr>
<td></td>
<td>1656</td>
<td>1674</td>
<td>1685</td>
<td>1689</td>
<td>1691</td>
<td>1704</td>
<td>1705</td>
</tr>
<tr>
<td></td>
<td>1722</td>
<td>1729</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETDA</td>
<td>715#</td>
<td>717</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETDAT</td>
<td>292</td>
<td>327</td>
<td>376</td>
<td>392</td>
<td>409#</td>
<td>426</td>
<td></td>
</tr>
<tr>
<td>GETEND</td>
<td>1916#</td>
<td>1920</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETFRM</td>
<td>311</td>
<td>380</td>
<td>396</td>
<td>441</td>
<td>458#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETINF</td>
<td>1632#</td>
<td>1644</td>
<td>1648</td>
<td>1665</td>
<td>1690</td>
<td>1702</td>
<td>1720</td>
</tr>
<tr>
<td>GETN</td>
<td>376</td>
<td>379#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETN2</td>
<td>355#</td>
<td>362</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETNUM</td>
<td>293</td>
<td>352#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETSWT</td>
<td>325</td>
<td>378</td>
<td>394</td>
<td>443</td>
<td>472</td>
<td>481#</td>
<td></td>
</tr>
<tr>
<td>GN2</td>
<td>393#</td>
<td>374</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GNAM</td>
<td>296#</td>
<td>322</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GORCK2</td>
<td>1297#</td>
<td>1349</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GOTA2</td>
<td>323</td>
<td>382</td>
<td>398</td>
<td>439</td>
<td>474</td>
<td>505</td>
<td>512#</td>
</tr>
<tr>
<td>GOFTRM</td>
<td>457</td>
<td>458#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GOFTRM</td>
<td>1600</td>
<td>1664</td>
<td>1668</td>
<td>1674#</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEADR</td>
<td>687</td>
<td>697#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBUF1</td>
<td>242#</td>
<td>609</td>
<td>830</td>
<td>927</td>
<td>1317</td>
<td>1319</td>
<td>1321</td>
</tr>
<tr>
<td>IBUF2</td>
<td>241#</td>
<td>1562</td>
<td>1603</td>
<td>1687</td>
<td>1879</td>
<td>1881</td>
<td></td>
</tr>
<tr>
<td>IBUFF</td>
<td>241#</td>
<td>1602</td>
<td>1609</td>
<td>1618</td>
<td>1878</td>
<td>1885</td>
<td></td>
</tr>
<tr>
<td>IGNOR1</td>
<td>1130</td>
<td>1154#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IGNOR2</td>
<td>1143</td>
<td>1154#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IGNORE</td>
<td>1153</td>
<td>1159#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>240#</td>
<td>256#</td>
<td>401</td>
<td>405</td>
<td>429#</td>
<td>460#</td>
<td></td>
</tr>
<tr>
<td>ILSWT</td>
<td>641#</td>
<td>507#</td>
<td>573</td>
<td>1455</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INAM2</td>
<td>214#</td>
<td>611</td>
<td>613</td>
<td>614</td>
<td>615</td>
<td>937</td>
<td>939</td>
</tr>
<tr>
<td>INCHR</td>
<td>242#</td>
<td>660</td>
<td>664</td>
<td>710</td>
<td>715</td>
<td>820</td>
<td>943</td>
</tr>
<tr>
<td>INPUT</td>
<td>1735#</td>
<td>1762</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INRD</td>
<td>241#</td>
<td>1627</td>
<td>1629</td>
<td>1632</td>
<td>1646</td>
<td>1659</td>
<td>1652</td>
</tr>
<tr>
<td>IODC</td>
<td>259#</td>
<td>773</td>
<td>786</td>
<td>797</td>
<td>824</td>
<td>814</td>
<td>839</td>
</tr>
<tr>
<td>KEEPFM</td>
<td>1711</td>
<td>1721#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESSND</td>
<td>841</td>
<td>847#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESSON</td>
<td>243#</td>
<td>271</td>
<td>317</td>
<td>568</td>
<td>570</td>
<td>647</td>
<td>718</td>
</tr>
<tr>
<td>LERM</td>
<td>1474</td>
<td>1472</td>
<td>1657</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIMI</td>
<td>253#</td>
<td>1162</td>
<td>1253</td>
<td>1279</td>
<td>1284</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIM2</td>
<td>253#</td>
<td>1162</td>
<td>1261</td>
<td>1218</td>
<td>1254</td>
<td>1280</td>
<td>1285</td>
</tr>
<tr>
<td>LRT</td>
<td>263#</td>
<td>399</td>
<td>866</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LRTTPT</td>
<td>249#</td>
<td>632</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASDEL</td>
<td>1931</td>
<td>1549#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TUPTOT</td>
<td>242#</td>
<td>628</td>
<td>808</td>
<td>899</td>
<td>1051</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>YNSF</td>
<td>1356</td>
<td>1362#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YRFL</td>
<td>933</td>
<td>936#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UAOH</td>
<td>1867#</td>
<td>1957</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNPNFD</td>
<td>1468</td>
<td>1469#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UP</td>
<td>1251#</td>
<td>1265</td>
<td>1275</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPD</td>
<td>1918</td>
<td>1925#</td>
<td>1929</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDT</td>
<td>253</td>
<td>1863#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WTOT</td>
<td>242#</td>
<td>626</td>
<td>795</td>
<td>889</td>
<td>905</td>
<td>913</td>
<td></td>
</tr>
<tr>
<td>YES1</td>
<td>1386</td>
<td>1390#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YES2</td>
<td>1787</td>
<td>1770#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YES3</td>
<td>1806</td>
<td>1809#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>124g#</td>
<td>1246</td>
<td>1257</td>
<td>1264</td>
<td>1271</td>
<td>1276</td>
<td>1293</td>
</tr>
</tbody>
</table>
LPTRPT  MACRO 47(294)-7 28141 2-AUG-74 PAGE 1

LPTRPT  MAC  6-JUN-74

TITLE LPTRPT

COMMENT %

WRITTEN BY CARRET A. VANDER LUST  COMPLETED 9-FEB-74

PART OF LRPRF.SHR, VERSION 3

LPTRPT.MAC PREPARES COMPLETE LEARN LESSON REPORTS AND STUDENT REPORTS. THE DATA FOR THE REPORTS IS OBTAINED FROM THE DISK FILE MASTER.REC. THE REPORT FILE IS WRITTEN ON THE DISK INTO A FILE WHOSE NAME WAS SELECTED BY RPRT.MAC (SECTION B). THE REPORT FILE IS THEN PLACED IN PRINT QUEUE SO THAT IT WILL BE PRINTED ON THE LINE PRINTER, (SEE RPRT.MAC FOR A DESCRIPTION OF THE DATA FORMAT IN MASTER.REC)

CONTENTS OF LPTRPT.MAC

SECTION A, WRITES THE REPORT HEADING

SECTION B, GETS SUMMARY DATA FROM MASTER.REC.

SECTION C, GETS FRAME DATA FROM MASTER.REC.

SECTION D, WRITES THE SUMMARY DATA ON THE DISK.

SECTION E, WRITES THE FRAME DATA ON THE DISK.

SECTION F, PLACES THE REPORT FILE IN PRINT QUEUE.

SECTION G, REINITIALIZE THE REPORT GENERATOR.

ENTRY POINT TO LPTRPT.MAC

LPTRPT - FROM RPRT.MAC (SECTION C), LOCATED IN SECTION A

EXTERNAL SUBROUTINES CALLED BY LPTRPT.MAC

INCHR - INPUTS A CHARACTER FROM THE DISK INTO AC=8, LOCATED IN AUX3.MAC (SECTION B)

IOD - OUTPUTS THE INTEGER IN AC=8 TO THE DISK, LOCATED IN AUX3.MAC (SECTION D)

OTDAT2 - OUTPUTS THE DATE IN AC=8 TO THE DISK, LOCATED IN AUX3.MAC (SECTION J)

OTTIM2 - OUTPUTS THE TIME IN AC=8 TO THE DISK, LOCATED IN AUX3.MAC (SECTION K)
OUTCHR - OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION F)

OUTDAT - OUTPUTS THE CURRENT DATE TO THE DISK, LOCATED IN AUX3.MAC (SECTION J)

OUTLES - OUTPUTS THE LESSON NAME IN AC-D TO THE DISK, LOCATED IN AUX3.MAC (SECTION I)

OUTPEN - OUTPUTS THE PROGRAMMER NO. IN AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION L)

OUTSTR - OUTPUTS A STRING OF CHARACTERS TO THE DISK, LOCATED IN AUX3.MAC (SECTION E)

OUTTIM - OUTPUTS THE CURRENT TIME TO THE DISK, LOCATED IN AUX3.MAC (SECTION K)

OUTTH - OUTPUTS LESSON TIME OR ANSWERING TIME FROM AC-A TO THE DISK, LOCATED IN AUX3.MAC (SECTION H)

RTJUST - RIGHT JUSTIFIES THE NUMBER IN AC-A, LOCATED IN AUX3.MAC (SECTION N)

RPRINT - PLACES THE REPORT FILE IN PRINT QUEUE, LOCATED IN RPRINT.MAC (SECTION A)

EXTERNAL MEMORY REFERENCES (ALL IN RPTLOW.MAC)

ABO - REPORT COMMAND SWITCH

ANSTOT - TOTAL ANSWERING TIME IS ACCUMULATED HERE

DATA - LIST OF SUMMARY DATA (16 WORDS)

DAYT - DATES SPECIFIED IN REPORT COMMAND (2 WORDS)

EXTOT - NO. OF TIMES LESSON WAS EXECUTED

FRAME - FRAME NO. SPECIFIED IN REPORT COMMAND (2 WORDS)

IBUF - DISK BUFFER HEADER (3 WORDS)

LESSON - LESSON NAME SPECIFIED IN REPORT COMMAND

NTTOT - NO. OF NEUTRAL RESPONSES IS ACCUMULATED HERE

PARAM - BEGINNING OF FRAME DATA LIST

PCS - RETURN ADDRESS IS STORED HERE
PPN1 = PROGRAMMER NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)

RESTOT = TOTAL NO. OF RESPONSES IS ACCUMULATED HERE

RTTOT = TOTAL NO. OF CORRECT RESPONSES IS ACCUMULATED HERE

RUNITM = TOTAL CPU TIME IS ACCUMULATED HERE

TIMTOT = TOTAL LESSON TIME IS ACCUMULATED HERE

TUPTOT = TOTAL NO. OF TIME UPS IS ACCUMULATED HERE

WGTOT = TOTAL NO. OF WRONG RESPONSES IS ACCUMULATED HERE

DESCRIPTION OF FLAGS STORED IN AC=0

BIT 25 REVIEW FRAME
BIT 26 STUDENT REPORT

% HISEG

ENTRY LPRTRPT

EXTERN TUNITOT,TUITOT,WGTOT,PARAM,FRAME
EXTERN ANSTOT,IOQ,NTTOT,OUTTH,RESTOT,RTJUST,RTTOT,RUNITM
EXTERN PCL,ABD,DAY,INCCH,LESSON,OTTIM,OUTIM,ODAT2,OUTDAT
EXTERN OUTCHR,OUTLES,OUTPPN,OUTSTR,PN1,EXTOT,RPRINT,DATA,IBUF

000000  FLAGS=0
000001  A#1
000002  B#2
000003  C#3
000004  D#4
000005  E#5
000006  F#6
000007  G#7
000017  Q#17
RECLST=PARAM+1
FMTDT=RECLST+1713

SECTION A. WRITE THE REPORT HEADING,

LPRTRPT MOVE B,ABD ;GET THE REPORT COMMAND SWITCH
CAIN B,ABD ;IS THIS A STUDENT REPORT?
YES, GO TO STORPT (SECTION B)
NO, CLEAR THE STUDENT REPORT FLAG
<table>
<thead>
<tr>
<th>Address</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000025</td>
<td>012 012 012 012 012</td>
<td>Lesson1</td>
</tr>
<tr>
<td>000026</td>
<td>015 012 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000027</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000028</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000029</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000030</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000031</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000032</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000033</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000034</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000035</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000036</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000037</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000038</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000039</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000040</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000041</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000042</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000043</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000044</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000045</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000046</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000047</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000048</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000049</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000050</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000051</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000052</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000053</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000054</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000055</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000056</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000057</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000058</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000059</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000060</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000061</td>
<td>015 010 101 122 116</td>
<td></td>
</tr>
<tr>
<td>000062</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>000063</td>
<td>040 040 040 040 040</td>
<td></td>
</tr>
</tbody>
</table>

The code appears to be a program for a computer system, including instructions for initializing, setting flags, and processing lessons. The program includes calls to `JSP Q.OUTSTR` and `JSP Q.OUTLE` for outputting strings and lesson names, respectively. The program also includes a call to `LEASHDG` for handling lesson names and a call to `MOVE D.LESSON` for getting the lesson name specified in the report command.
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>352</td>
<td>003357</td>
<td>305 03 02 30240</td>
</tr>
<tr>
<td>353</td>
<td>003376</td>
<td>265 17 00 002160*</td>
</tr>
<tr>
<td>354</td>
<td>003371</td>
<td>305 03 00 002420*</td>
</tr>
<tr>
<td>355</td>
<td>003372</td>
<td>265 17 00 002378*</td>
</tr>
<tr>
<td>356</td>
<td>003373</td>
<td>305 03 00 002750</td>
</tr>
<tr>
<td>357</td>
<td>003374</td>
<td>265 17 00 002372*</td>
</tr>
<tr>
<td>358</td>
<td>003379</td>
<td>200 01 00 002023</td>
</tr>
<tr>
<td>359</td>
<td>003376</td>
<td>231 01 00 001750</td>
</tr>
<tr>
<td>360</td>
<td>003377</td>
<td>222 02 00 002263*</td>
</tr>
<tr>
<td>361</td>
<td>002400</td>
<td>265 17 00 002358*</td>
</tr>
<tr>
<td>362</td>
<td>002451</td>
<td>201 01 00 002021</td>
</tr>
<tr>
<td>363</td>
<td>002402</td>
<td>265 17 00 002374</td>
</tr>
<tr>
<td>364</td>
<td>002433</td>
<td>230 01 00 003377*</td>
</tr>
<tr>
<td>365</td>
<td>002454</td>
<td>231 01 00 002021</td>
</tr>
<tr>
<td>366</td>
<td>002456</td>
<td>307 00 00 002358*</td>
</tr>
<tr>
<td>367</td>
<td>002480</td>
<td>350 00 00 002021</td>
</tr>
<tr>
<td>368</td>
<td>002481</td>
<td>265 17 00 002358*</td>
</tr>
</tbody>
</table>
| 369  | 002438 | 265 17 00 002358* | JSP Q.OUTSTR | ASC12 /
| 370  | 002411 | 201 02 00 002021 | MOVE B+ABO | GET THE NO. OF DATA WORDS FOR THIS LESSON |
| 371  | 002412 | 201 02 00 002021 | SUB1 B.14 | SUBTRACT OFF NO. OF WORDS OF SUMMARY DATA |
| 372  | 002413 | 201 02 00 002021 | MOVE B+ABO | PUT RESULT BACK IN ABO |

---

### SECTION C. GET THE FRAME DATA FROM MASTER,REG.

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>378</td>
<td>002415</td>
<td>370 02 00 002021</td>
</tr>
<tr>
<td>381</td>
<td>002416</td>
<td>371 02 00 002021</td>
</tr>
<tr>
<td>382</td>
<td>002417</td>
<td>265 17 00 002021</td>
</tr>
<tr>
<td>384</td>
<td>002420</td>
<td>202 01 00 002021</td>
</tr>
<tr>
<td>385</td>
<td>002421</td>
<td>265 17 00 002021</td>
</tr>
<tr>
<td>386</td>
<td>002422</td>
<td>370 00 00 002021</td>
</tr>
<tr>
<td>388</td>
<td>002423</td>
<td>202 01 00 002021</td>
</tr>
<tr>
<td>389</td>
<td>002424</td>
<td>400 03 00 002021</td>
</tr>
<tr>
<td>390</td>
<td>002425</td>
<td>135 01 00 002021</td>
</tr>
<tr>
<td>391</td>
<td>002426</td>
<td>305 01 00 002021</td>
</tr>
<tr>
<td>392</td>
<td>002427</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>393</td>
<td>002428</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>394</td>
<td>002429</td>
<td>371 01 00 002021</td>
</tr>
<tr>
<td>395</td>
<td>002430</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>396</td>
<td>002431</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>397</td>
<td>002432</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>398</td>
<td>002433</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>399</td>
<td>002434</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>400</td>
<td>002435</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>401</td>
<td>002436</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>402</td>
<td>002437</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>403</td>
<td>002438</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>404</td>
<td>002439</td>
<td>315 01 00 002021</td>
</tr>
<tr>
<td>405</td>
<td>002440</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>406</td>
<td>002441</td>
<td>254 02 00 002021</td>
</tr>
<tr>
<td>407</td>
<td>002442</td>
<td>326 02 00 002021</td>
</tr>
<tr>
<td>408</td>
<td>002443</td>
<td>211 03 00 002021</td>
</tr>
<tr>
<td>409</td>
<td>002444</td>
<td>211 03 00 002021</td>
</tr>
<tr>
<td>410</td>
<td>002445</td>
<td>221 03 00 002021</td>
</tr>
</tbody>
</table>
### SECTION D. WRITE THE SUMMARY DATA ON THE DISK

<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>467</td>
<td><code>002537</code> 265 17 0 20 000410 END FIL JSP Q;OUTSTR ;OUTPUT THE FOLLOWING STRING ASCIE /</td>
</tr>
<tr>
<td></td>
<td>NO. TIMES LESSON WAS EXECUTED /</td>
</tr>
</tbody>
</table>

### Summary Data

<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>470</td>
<td><code>002542</code> 125 12 0 12 04 03</td>
</tr>
<tr>
<td>471</td>
<td><code>002543</code> 124 11   0 12 04 04</td>
</tr>
<tr>
<td>472</td>
<td><code>002544</code> 123 10   0 12 04 04</td>
</tr>
<tr>
<td>473</td>
<td><code>002545</code> 122 10   0 12 04 04</td>
</tr>
<tr>
<td>474</td>
<td><code>002546</code> 121 10   0 12 04 04</td>
</tr>
<tr>
<td>475</td>
<td><code>002547</code> 120 10   0 12 04 04</td>
</tr>
</tbody>
</table>

### True Flags

<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>476</td>
<td><code>002552</code> 015 02 015 012 043</td>
</tr>
<tr>
<td>477</td>
<td><code>002553</code> 040 04 040 040 040</td>
</tr>
<tr>
<td>478</td>
<td><code>002554</code> 100 00   0 00 00 00</td>
</tr>
<tr>
<td>479</td>
<td><code>002555</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>480</td>
<td><code>002556</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>481</td>
<td><code>002557</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>482</td>
<td><code>002558</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>483</td>
<td><code>002559</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>484</td>
<td><code>002560</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>485</td>
<td><code>002561</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>486</td>
<td><code>002562</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>487</td>
<td><code>002563</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>488</td>
<td><code>002564</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>489</td>
<td><code>002565</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>490</td>
<td><code>002566</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>491</td>
<td><code>002567</code> 040 10 040 105 120 105</td>
</tr>
</tbody>
</table>

### Elapsed Time

<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>492</td>
<td><code>002568</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>493</td>
<td><code>002569</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>494</td>
<td><code>002570</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>495</td>
<td><code>002571</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>496</td>
<td><code>002572</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>497</td>
<td><code>002573</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>498</td>
<td><code>002574</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>499</td>
<td><code>002575</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>500</td>
<td><code>002576</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>501</td>
<td><code>002577</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>502</td>
<td><code>002578</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>503</td>
<td><code>002579</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>504</td>
<td><code>002580</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>505</td>
<td><code>002581</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>506</td>
<td><code>002582</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>507</td>
<td><code>002583</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>508</td>
<td><code>002584</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>509</td>
<td><code>002585</code> 040 10 040 105 120 105</td>
</tr>
<tr>
<td>510</td>
<td><code>002586</code> 040 10 040 105 120 105</td>
</tr>
</tbody>
</table>

### Summary

- ** avenues of data words for this lesson.**
- **subtract no. of words of summary data.**
- **got to fnosa (section 6).**
- **write the summary data on the disk.**
- **output the following string ascie /**
- **no. times lesson was executed /**
- **is this a student report?**
- **true flags, 100.**
- **ave. elapsed time /**
- **move a, timtot, get the total elapsed time.**
- **divide by no. of times lesson was executed.**
- **get ac to zero for the output subroutine.**
- **output the elapsed time.**
- **output the following string ascie / minutes.**
- **ave. total answering time /**
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>518</td>
<td>202761</td>
<td>MOVE A, R</td>
<td>MOVE A, RTOT 1 GET THE TOTAL NO. OF CORRECT ANSWERS</td>
</tr>
<tr>
<td>519</td>
<td>202674</td>
<td>JSP G, 100</td>
<td>JSP G, 100 1 OUTPUT NO. OF CORRECT ANSWERS</td>
</tr>
<tr>
<td>520</td>
<td>202675</td>
<td>JSP G, OUTSTR</td>
<td>JSP G, OUTSTR 1 OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>521</td>
<td>202676</td>
<td>ASCII /</td>
<td>TOTAL NO. OF RESPONSES /</td>
</tr>
<tr>
<td>522</td>
<td>202677</td>
<td>MOVE A, HGTOT 1 GET THE TOTAL NO. OF WRONG ANSWERS</td>
<td></td>
</tr>
<tr>
<td>523</td>
<td>202678</td>
<td>JSP G, 100</td>
<td>JSP G, 100 1 OUTPUT NO. OF WRONG ANSWERS</td>
</tr>
<tr>
<td>524</td>
<td>202679</td>
<td>JSP G, OUTSTR</td>
<td>JSP G, OUTSTR 1 OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>525</td>
<td>202680</td>
<td>ASCII /</td>
<td>TOTAL NO. OF NEUTRAL RESPONSES /</td>
</tr>
<tr>
<td>526</td>
<td>202681</td>
<td>MOVE A, R</td>
<td>MOVE A, RTOT 1 GET THE TOTAL NO. OF NEUTRAL ANSWERS</td>
</tr>
<tr>
<td>527</td>
<td>202682</td>
<td>JSP G, 100</td>
<td>JSP G, 100 1 OUTPUT NO. OF NEUTRAL ANSWERS</td>
</tr>
<tr>
<td>528</td>
<td>202683</td>
<td>JSP G, OUTSTR</td>
<td>JSP G, OUTSTR 1 OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>529</td>
<td>202684</td>
<td>ASCII /</td>
<td>TOTAL NO. OF TIME UPS /</td>
</tr>
<tr>
<td>530</td>
<td>202685</td>
<td>MOVE A, TUPTOT 1 GET THE TOTAL NO. OF TIME UPS</td>
<td></td>
</tr>
<tr>
<td>531</td>
<td>202686</td>
<td>JSP G, 100</td>
<td>JSP G, 100 1 OUTPUT NO. OF TIME UPS</td>
</tr>
<tr>
<td>532</td>
<td>202687</td>
<td>JSP G, OUTSTR</td>
<td>JSP G, OUTSTR 1 OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>533</td>
<td>202688</td>
<td>ASCII /</td>
<td>ERROR RATE /</td>
</tr>
<tr>
<td>534</td>
<td>202689</td>
<td>MOVE B, R</td>
<td>MOVE B, RTOT 1 GET THE NO. OF CORRECT ANSWERS</td>
</tr>
<tr>
<td>535</td>
<td>202690</td>
<td>ADD B, HGTOT 1 ADD ON THE NO. OF WRONG ANSWERS</td>
<td></td>
</tr>
<tr>
<td>536</td>
<td>202691</td>
<td>MOVE A, HGTOT 1 INPUT NO. OF WRONG ANSWERS IN AC-A</td>
<td></td>
</tr>
<tr>
<td>537</td>
<td>202692</td>
<td>IMULI A, 0010000</td>
<td>IMULI A, 0010000 1 SHIFT NO. OVER 4 PLACES</td>
</tr>
<tr>
<td>538</td>
<td>202693</td>
<td>IDIV A, B</td>
<td>IDIV A, B 1 COMPUTE THE PERCENTAGE OF WRONG ANSWERS</td>
</tr>
<tr>
<td>Line</td>
<td>Address</td>
<td>Contents</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>543</td>
<td>02754'</td>
<td>231 01 0 000012 IDIVI A, +010</td>
<td></td>
</tr>
<tr>
<td>544</td>
<td>02755'</td>
<td>301 02 0 000005 CALL B, 5</td>
<td></td>
</tr>
<tr>
<td>545</td>
<td>02756'</td>
<td>350 00 0 000001 AOS A</td>
<td></td>
</tr>
<tr>
<td>546</td>
<td>02757'</td>
<td>231 01 0 000012 IDIVI A, +010</td>
<td></td>
</tr>
<tr>
<td>547</td>
<td>02758'</td>
<td>302 02 0 000751 MOVEM B, WQTOTatables the fractional part</td>
<td></td>
</tr>
<tr>
<td>548</td>
<td>02759'</td>
<td>265 17 0 000737 JSP Q, 100</td>
<td></td>
</tr>
<tr>
<td>549</td>
<td>02760'</td>
<td>265 17 0 00056 JSP Q, OUTCHR</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td>02761'</td>
<td>265 17 0 000637 JSP Q, OUTCHR</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td>02762'</td>
<td>265 17 0 000761 JSP Q, OUTSTR</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td>02763'</td>
<td>265 17 0 000748 JSP Q, OUTSTR</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td>02764'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>554</td>
<td>02765'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>555</td>
<td>02766'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>556</td>
<td>02767'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>557</td>
<td>02768'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>558</td>
<td>02769'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>559</td>
<td>02770'</td>
<td>040 045 019 012 012 /</td>
<td></td>
</tr>
<tr>
<td>560</td>
<td>02771'</td>
<td>265 17 0 000766 JSP Q, OUTSTR</td>
<td></td>
</tr>
<tr>
<td>561</td>
<td>02772'</td>
<td>315 012 015 012 040</td>
<td></td>
</tr>
<tr>
<td>562</td>
<td>02773'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>563</td>
<td>02774'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>564</td>
<td>02775'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>565</td>
<td>02776'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>566</td>
<td>02777'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>02778'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>568</td>
<td>02779'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>569</td>
<td>02780'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>570</td>
<td>02781'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>571</td>
<td>02782'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>572</td>
<td>02783'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>573</td>
<td>02784'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>574</td>
<td>02785'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>575</td>
<td>02786'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>576</td>
<td>02787'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>577</td>
<td>02788'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>578</td>
<td>02789'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>579</td>
<td>02790'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>580</td>
<td>02791'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>581</td>
<td>02792'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>582</td>
<td>02793'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>583</td>
<td>02794'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>584</td>
<td>02795'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>585</td>
<td>02796'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>586</td>
<td>02797'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>587</td>
<td>02798'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>588</td>
<td>02799'</td>
<td>340 040 040 040 040</td>
<td></td>
</tr>
<tr>
<td>590</td>
<td>02800'</td>
<td>620 00 0 000000 END2: 2020, CLEAR REVIEW FRAME FLAG</td>
<td></td>
</tr>
</tbody>
</table>

---

SECTION C. WRITE THE FRAME DATA ON THE DISK.

1. THE ROUTINE BELOW SORTS THE RECORD LIST IN ORDER OF INCREASING FRAME NO.

2. SETZB C, 0
3. HRRZ A, RECLST(C)
4. JSP R, END1
5. SETZB C, 0
6. HRRZ A, RECLST(C)
7. JUMPE B, END2
8. MOVEM B, WQTOT
9. MOVEM B, WQTOT
10. MOVEM B, WQTOT
I. REVIEW FRAME DATA

I. NOTREV: JSP G; OUTSTR :OUTPUT THE FOLLOWING STRING

ASCI/ 

I. FRAME:

I. MOVE A,C INPUT FRAME NO. IN AC-A

I. JSP G;IOD :OUTPUT THE FRAME NO.

I. JSP G;OUTSTR :OUTPUT THE FOLLOWING STRING

ASCI/ TYPE = I

I. LDB A,POINT 7,RECLST(F),6 IGET THE FRAME TYPE

I. JSP G;OUTCHR :OUTPUT FRAME TYPE

I. JSP G;OUTSTR :OUTPUT THE FOLLOWING STRING

ASCI/ 

I. NO. OF TIMES FRAME WAS EXECUTED:

I. LDB G,POINT 11,RECLST(F),17 IGET THE DATA INDEX FROM THE RECORD LIST

I. IMULI G,16 IMULTIPLY BY NO. OF DATA WORDS FOR EACH FRAME

I. HRR A,FMDAT(S) IGET NO. OF TIMES FRAME WAS EXECUTED

I. MOVE A,PARAM :SAVE IT

I. JSP G;IOD :OUTPUT NO. OF TIMES FRAME WAS EXECUTED
<table>
<thead>
<tr>
<th>LPTRPT</th>
<th>MACRO 47(284-1)-7 00141 2-AUG-74 PAGE 1-17</th>
</tr>
</thead>
<tbody>
<tr>
<td>001251</td>
<td>124 161 114 040 116</td>
</tr>
<tr>
<td>001252</td>
<td>124 161 114 040 116</td>
</tr>
<tr>
<td>001253</td>
<td>107 056 045 117 106</td>
</tr>
<tr>
<td>001254</td>
<td>040 124 111 115 109</td>
</tr>
<tr>
<td>001284</td>
<td>124 128 125 122 072</td>
</tr>
<tr>
<td>001294</td>
<td>040 000 000 020 086</td>
</tr>
<tr>
<td>694</td>
<td>001256</td>
</tr>
<tr>
<td>695</td>
<td>001257</td>
</tr>
<tr>
<td>696</td>
<td>001258</td>
</tr>
<tr>
<td>697</td>
<td>001259</td>
</tr>
<tr>
<td>698</td>
<td>001260</td>
</tr>
<tr>
<td>699</td>
<td>001261</td>
</tr>
<tr>
<td>001262</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001263</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001264</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001265</td>
<td>033 161 0 0 001236</td>
</tr>
<tr>
<td>001266</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001267</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001268</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001269</td>
<td>040 040 040 040 040</td>
</tr>
<tr>
<td>001270</td>
<td>hibited without permission. Further reproduction prohibited without permission.</td>
</tr>
</tbody>
</table>
NO ERRORS DETECTED

PROGRAM BREAK IS 801442

3K CORE USED
TITLE RPRINT

COMMENT %

PREPARED BY GARRET A. VANDER LUYT  COMPLETED 4-FEB-74

PART OF LRPRT.SHR  VERSION 3

SPECIAL NOTE! THIS SUBROUTINE IS A MODIFIED VERSION OF A SUB-
ROUTINE WHICH WAS WRITTEN BY NORMAN D. GRANT AT THE COMPUTER CENTER,
WESTERN MICHIGAN UNIVERSITY.

RPRINT.MAC PLACES REPORT LISTINGS IN PRINT QUEUE SO THAT THEY WILL
BE PRINTED ON THE LINE PRINTER.

CONTENTS OF RPRINT.MAC

SECTION A. SET UP QUEuing PARAMETERS,  

SECTION B. OUTPUT QUEuing PARAMETERS TO THE DISK,

SECTION C. WAKE UP THE SPOOLING PROGRAM,

ENTRY POINT TO RPRINT.MAC

RPRINT - FROM RPT.MAC (SECTIONS D AND F) ANDLPTRPT.MAC (SECTION F),
LOCATED IN SECTION A

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN RPLLOW,MAC)

EXTOT (#THSJOB) = USER’S JOB NUMBER

RUNTIM (#QUEDIR) = NAME OF QUEUING UID

INAM2 (#OPNBLK) = ARGUMENTS FOR OPEN UID (3 WORDS)

LESSON (#FILNAM) = FILENAME OF PARAMETER FILE

CMDSLST (#OLIST) = OUTPUT COMMAND LIST (2 WORDS)

DATA (#QLSPEC) = EXTENDED LOOKUP ARGUMENTS (14 WORDS)

DNAM = CONTAINS NAME OF FILE TO BE QUEUED

PARAM = LIST OF QUEuing PARAMETERS (32 WORDS)

HISEG
ENTRY RPRINT

EXTERN EXTOT, RUNITM, INAM2, LESSON, CMDSLT, DATA, ONAM, PARAH

THSJESJ=EXTOT
QEDIR=RUNITM
OPWBLK=INAM2
FILNAM=LESSON
OLIST=CMDSLT
QLSCPE=DATA

A=1
B=2
C=3
D=4
E=5
F=6
G=7
H=10
P=15
Q=17

SECTION A. SET UP QUEUING PARAMETERS IN CORE.

OPN = 16

SIXBIT / DSK/

MOVE A,[34,F111]

GETTAB A; ;GET MONITOR VERSION NO.

HALT

CAIGE A,50300; ;IS MONITOR VERSION 5.83 OR LATER?

HALT

OPEN 0,0PN

MOVE A,40

SETAM PARAH=1(A) ; CLEAR QUEUING PARAMETERS

SOUG A=-1

MOVE A,16

MOVE A,QLSPEC-1 MOVE 14 TO QLSCPE FOR EXTENDED LOOKUP

SETAM QLSCPE=1 100 LOOKUP IN USER'S OWN UPD

MOVE A,ONAH 1 GET FILENAME OF LIST FILE

MOVE A,QLSPEC-2 1 MOVE IT TO LOOKUP EFFECTIVE ADDRESS

MOVE A,PARAH+5 1 MOVE IT TO PARAMETER LIST ALSO

OPEN 0,OPEN 1 OPEN CHANNEL 1, DUMP MODE

LOOKUP 0,QLSCPE 1 DO LOOKUP ON LIST FILE
ISECTION B. WRITE PARAMETER FILE ON THE DISK.

MOVE A,[15,16]
GETTAB A;
GET FILE STRUCTURE CONTAINING SPOOLING UFD
MOVSI A,DISK
MOVE B,[6,16]
GETTAB B;
GET PPN FOR SPOOLING PROGRAMS
MOVE C,E00000,171
GETTAB B;
GET PPN FOR DEVICE SYS
MOVE D,C1,163
GETTAB B;
GET PPN FOR LINE PRINTER
MOVE E,C1,163
MOVE B,QUEDIR
MOVE C,OPNBLK
SETM OPNBLK=2
MOVEM A,OPNBLK=1
OPEN B,OPNBLK
HALT
MOVSI B,LPT1
GET STATION NO. OF LINE PRINTER
MOVE B,LPT1
SETB B
HRLI B,LPT1
MOVEM B,PAM=3
IFIND SUITABLE NAME FOR PARAMETER FILE

MOVEM B,DISK
IDIVI A,$100
CREDI1 MOVSI B,LPT1
MOVE C,POINT 6,B4,11
ADD A,THSB08
MOVE D,A
CREDI1 IDIVI D,$100
ADDI E,10
IPDB E,0
TLINE C,(7785) IDONE YET?
JST CREDI1 INO
MOVEM C,QUEUE
MOVEM B,FILNAM ISAVE FILENAME
MOVE E,QUEUE
TITLE AUX3

COMMENT %

WRITTEN BY GARRET A. VANDER LUGTCOMPLETED 5-FEB-74

PART OF LRPT, SHR; VERSION 3

AUX3, MAC CONTAINS A NUMBER OF SUBROUTINES WHICH ARE USED BY
PROGRAMS IN THE LRPT HIGH SEGMENT.

CONTENTS OF AUX3, MAC

SECTION A. INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE
END OF A LINE IS REACHED.

SECTION B. INPUTS A CHARACTER FROM THE DISK.

SECTION C. INPUTS AN INTEGER FROM A USER'S TERMINAL.

SECTION D. OUTPUTS AN INTEGER TO THE DISK.

SECTION E. OUTPUTS A STRING OF CHARACTERS TO THE DISK.

SECTION F. OUTPUTS A CHARACTER TO THE DISK.

SECTION G. INPUTS A DATA WORD FROM THE DISK.

SECTION H. OUTPUTS A DATA WORD TO THE DISK.

SECTION I. OUTPUTS A LESSON NAME TO THE DISK.

SECTION J. OUTPUTS A DATE TO THE DISK.

SECTION K. OUTPUTS TIME OF DAY TO THE DISK.

SECTION L. OUTPUTS A PROGRAMMER NUMBER TO THE DISK.

SECTION M. OUTPUTS LESSON AND ANSWERING TIMES TO THE DISK.

SECTION N. RIGHT JUSTIFIES NUMBERS IN A COLUMN.

ENTRY POINTS TO AUX3, MAC

EOLIN = FROM LRPT, MAC, LOCATED IN SECTION A

II = FROM RPRT, MAC, LOCATED IN SECTION C

II2 = FROM RPRT, MAC, LOCATED IN SECTION C
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>INCH - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION B</td>
</tr>
<tr>
<td>55</td>
<td>INRD - FROM RPRT.MAC, LOCATED IN SECTION G</td>
</tr>
<tr>
<td>56</td>
<td>IOO - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION D</td>
</tr>
<tr>
<td>57</td>
<td>OUTDAT2 - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION J</td>
</tr>
<tr>
<td>58</td>
<td>OTTIM2 - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION K</td>
</tr>
<tr>
<td>59</td>
<td>OUTCHR - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION F</td>
</tr>
<tr>
<td>60</td>
<td>OUTDAT - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION J</td>
</tr>
<tr>
<td>61</td>
<td>OUTLES - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION I</td>
</tr>
<tr>
<td>62</td>
<td>OUTP PN - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION L</td>
</tr>
<tr>
<td>63</td>
<td>OUTSTR - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION E</td>
</tr>
<tr>
<td>64</td>
<td>OUTTIM - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION K</td>
</tr>
<tr>
<td>65</td>
<td>OUTWD - FROM RPRT.MAC AND LPTRPT.MAC, LOCATED IN SECTION N</td>
</tr>
<tr>
<td>66</td>
<td>RTJUST - FROM RPRT.MAC, LOCATED IN SECTION N</td>
</tr>
</tbody>
</table>

**EXTERNAL MEMORY REFERENCES (ALL LOCATED IN RPTLOW.MAC)**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>67</td>
<td>IBUF - BUFFER HEADER FOR DISK INPUT (3 WORDS)</td>
</tr>
<tr>
<td>68</td>
<td>IBUF2 - BUFFER HEADER FOR DISK INPUT (3 WORDS)</td>
</tr>
<tr>
<td>69</td>
<td>QBUF - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)</td>
</tr>
<tr>
<td>70</td>
<td>QBUF2 - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)</td>
</tr>
<tr>
<td>71</td>
<td>PC2 - RETURN ADDRESS</td>
</tr>
<tr>
<td>72</td>
<td>POLST - PUSH DOWN LIST FOR STORING ASCII DIGITS OF A NUMBER BEING CONVERTED FROM BINARY TO ASCII CODE (10 WORDS)</td>
</tr>
</tbody>
</table>

**SEG**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>WISEG</td>
</tr>
<tr>
<td>103</td>
<td>ENTRY INCHR, IOO, OUTCHR, INRD, OUTWD, II, OUTSTR, I12, COLIN</td>
</tr>
<tr>
<td>104</td>
<td>INTERN OUTLES, OUTDAT, OUTDAT, OTTIM, OUTTIM, OUTP PN, OUTWD, RTJUST</td>
</tr>
<tr>
<td>105</td>
<td>EXtern IBUF, POLST, QBUF, IBUF2, QBUF2, PC2</td>
</tr>
</tbody>
</table>
### SECTION A. THIS SUBROUTINE INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL AN END OF LINE IS ENCOUNTERED.

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>000201 051 04 0 0 0 000201</td>
<td>TDCALL 4.A INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>121</td>
<td>000201 302 01 0 0 00012</td>
<td>EOLINI CALE A:12 IEND OF LINE</td>
</tr>
<tr>
<td>122</td>
<td>000221 254 00 0 0 000208</td>
<td>JIRST ++2 INC. GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>123</td>
<td>000331 254 00 0 17 000200</td>
<td>JIRST (Q) IYES, RETURN</td>
</tr>
</tbody>
</table>

### SECTION B. THIS SUBROUTINE INPUTS A CHARACTER FROM THE DISK CHANNEL 1. THE CHARACTER IS PLACED IN AC-A.

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>130</td>
<td>000204 375 00 0 0 0 000028</td>
<td>INCHR SOSGE IBUF=2 IDECREMENT CHARACTER COUNT</td>
</tr>
<tr>
<td>131</td>
<td>000205 254 00 0 0 000018</td>
<td>JIRST GETBUF IIF BUFFER IS EMPTY, GO GET ANOTHER BUFFER</td>
</tr>
<tr>
<td>132</td>
<td>000206 134 01 0 0 000018</td>
<td>GETNX1 ILDS A:1BUF=1 IPUT CHARACTER IN AC-A</td>
</tr>
<tr>
<td>133</td>
<td>000207 254 00 0 17 000000</td>
<td>JIRST (Q) IRETURN</td>
</tr>
<tr>
<td>134</td>
<td>000208 056 01 0 0 000000</td>
<td>GETBUF IN 1, IF FULL BUFFER</td>
</tr>
<tr>
<td>135</td>
<td>000209 254 00 0 0 000048</td>
<td>JIRST INCHR IIF NO ERROR, GO GET NEXT CHARACTER</td>
</tr>
<tr>
<td>136</td>
<td>00020A 063 01 0 0 020000</td>
<td>STAT2 1,20000 IEND OF FILE?</td>
</tr>
<tr>
<td>137</td>
<td>00020B 254 00 0 17 000000</td>
<td>JIRST (Q) IYES, RETURN</td>
</tr>
<tr>
<td>138</td>
<td>TDCALL 3;CASQ1 I DISK INPUT ERROR ON CHANNEL 1 + AUX3</td>
<td></td>
</tr>
<tr>
<td>139</td>
<td>000141 091 03 0 0 000371</td>
<td>HALT</td>
</tr>
<tr>
<td>140</td>
<td>000015 254 04 0 0 000000</td>
<td></td>
</tr>
</tbody>
</table>

### SECTION C. THIS SUBROUTINE INPUTS AN INTEGER FROM A USER'S TERMINAL AND PLACES IT IN AC-B. IF THE 112 ENTRY POINT IS USED, LEADING SPACES ARE IGNORED.

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>149</td>
<td>000161 400 02 0 0 000000</td>
<td>SETZ B.</td>
</tr>
<tr>
<td>150</td>
<td>000171 251 04 0 0 000001</td>
<td>TDCALL 4.A INPUT CHARACTER</td>
</tr>
<tr>
<td>151</td>
<td>000201 301 01 0 0 000000</td>
<td>CALA A:60</td>
</tr>
<tr>
<td>152</td>
<td>000201 365 01 0 0 000000</td>
<td>CALE A:61</td>
</tr>
<tr>
<td>153</td>
<td>000221 254 00 0 17 000000</td>
<td>JIRST (Q) IRETURN IF CHARACTER IS NOT A DIGIT</td>
</tr>
<tr>
<td>154</td>
<td>000231 221 22 0 0 000012</td>
<td>IMUL B:12 IMULTIPLY BY 10</td>
</tr>
<tr>
<td>155</td>
<td>000241 271 02 0 0 777720</td>
<td>ADDI B:60(A) IADD VALUE OF DIGIT TO NO. IN AC-B</td>
</tr>
<tr>
<td>156</td>
<td>000251 254 00 0 0 000171</td>
<td>JIRST II+1 IGO GET NEXT DIGIT</td>
</tr>
<tr>
<td>157</td>
<td>000261 055 04 0 0 000001</td>
<td>TDCALL 4.A IINPUT CHARACTER</td>
</tr>
<tr>
<td>158</td>
<td>000271 306 01 0 0 000348</td>
<td>CALA A:60 IIS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>159</td>
<td>000361 254 00 0 0 000261</td>
<td>JIRST ++2 IYES, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>107#</td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>352#</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>109#</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>110#</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>111#</td>
</tr>
<tr>
<td></td>
<td>EOLIN</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>112#</td>
</tr>
<tr>
<td></td>
<td>G</td>
<td>113#</td>
</tr>
<tr>
<td></td>
<td>GETBUF</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td>GETNXT</td>
<td>132#</td>
</tr>
<tr>
<td></td>
<td>GTBUF</td>
<td>215</td>
</tr>
<tr>
<td></td>
<td>GTNXT</td>
<td>216#</td>
</tr>
<tr>
<td></td>
<td>IBUF</td>
<td>125#</td>
</tr>
<tr>
<td></td>
<td>IBUF2</td>
<td>125#</td>
</tr>
<tr>
<td></td>
<td>II2</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>INCHR</td>
<td>122#</td>
</tr>
<tr>
<td></td>
<td>INRD</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>IOD</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>OBUF</td>
<td>105#</td>
</tr>
<tr>
<td></td>
<td>OBUF2</td>
<td>105#</td>
</tr>
<tr>
<td></td>
<td>ODTAM2</td>
<td>105#</td>
</tr>
<tr>
<td></td>
<td>OUTCHR</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>OUTDAT</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td>OUTLES</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>OUTPN</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>OUTSTR</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>OUTTIM</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>OUTFM</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>OUTURD</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>PCE</td>
<td>105#</td>
</tr>
<tr>
<td></td>
<td>POLST</td>
<td>105#</td>
</tr>
<tr>
<td></td>
<td>PTOBUF</td>
<td>233</td>
</tr>
<tr>
<td></td>
<td>PTVNX</td>
<td>234#</td>
</tr>
<tr>
<td></td>
<td>PTPUT</td>
<td>199</td>
</tr>
<tr>
<td></td>
<td>PUTNBX</td>
<td>233#</td>
</tr>
<tr>
<td></td>
<td>Q</td>
<td>114#</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>248</td>
</tr>
<tr>
<td></td>
<td>T</td>
<td>309</td>
</tr>
<tr>
<td></td>
<td>RTJUST</td>
<td>103</td>
</tr>
</tbody>
</table>
RPTLOW MACRO 47(204)-7 08147 2-AUG-74 PAGE 1

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

TITLE RPTLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUST COMPLETED 6-FEB-74

PART OF LRPT, SHR, VERSION 3

RPTLOW, MAC IS A DUMM ) LOW SEGMENT WHICH IS USED TO LABEL MEMORY
LOCATIONS IN THE LOW SEGMENT SO THAT THEY CAN BE EASILY REFERENCED
BY PROGRAMS IN THE LRPT HIGH SEGMENT, SINCE NO INSTRUCTIONS OR
CONSTANTS ARE STORED IN THE LOW SEGMENT, NO BINARY CORE IMAGE OF IT
IS PRODUCED BY THE SAVE MONITOR COMMAND.

ENTRY IBUFF, OBUFF, INAM2, DONM2, IBUFF2, OBUFF2, PARAM, ARGINT, FRAME
INTERN IBUFF, OBUFF, TRBLLK, JOBN0, LIM1, LIM2
INTERN POLST, DAT1, EXT1, TIM1, ANST1, REST1, RUN1
INTERN TKT1, LST1, TUP1, LORG, DEV1, PNH, MONTH, DAY1, ABQ
INTERN INAM1, INAM, IBUFF, OBUFF, PCH1, PCH2, STORE, PNM, SM.ACT1, STOT

080137
LOC JOBER
320137
000311 000001
000001

STORE1 BLOCK 1 USED BY LRPT, MAC
ARGINT BLOCK 2 USED BY LRPT, MAC
JOBN0 BLOCK 3 JOB NUMBER
TRBLLK BLOCK 4 INTERRUPT Block IS STORED HERE
INLOCI BLOCK 5 INTERRUPT ROUTINE IS STORED HERE
IBUFF BLOCK 6 USED BY RPRR, MAC AND LPRR, MAC AND AUX, MAC
OBUFF BLOCK 7 USED BY RPRR, MAC AND AUX, MAC
INAM BLOCK 8 USED BY RPRR, MAC
ONAM BLOCK 9 USED BY RPRR, MAC AND RPRINT, MAC
PPNI BLOCK 10 USER'S PROJECT-PROGRAMMER NO.
INBUFF BLOCK 11 USED BY RPRR, MAC AND LPRR, MAC (CHANNEL 1 INPUT BUFFER)
OUTBUFF BLOCK 12 USED BY RPRR, MAC AND LPRR, MAC (CHANNEL 1 OUTPUT BUFFER)

ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN LLOW1, MAC.

IBUFF1 BLOCK 233 USED BY RPRR, MAC (CHANNEL 1 INPUT BUFFER)
OBUFF1 BLOCK 234 USED BY RPRR, MAC (CHANNEL 1 OUTPUT BUFFER)
IBUFF2 BLOCK 233 USED BY RPRR, MAC AND AUX3, MAC
OBUFF2 BLOCK 234 USED BY RPRR, MAC AND AUX3, MAC
INAM2 BLOCK 235 USED BY RPRR, MAC AND RPRINT, MAC
ONAM2 BLOCK 236 USED BY RPRR, MAC
PCH1 BLOCK 237 USED BY RPRR, MAC AND LPRR, MAC
PCH2 BLOCK 238 USED BY RPRR, MAC AND AUX3, MAC
CMDLST BLOCK 239 USED BY RPRR, MAC AND RPRINT, MAC

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
NO ERRORS DETECTED

PROGRAM BREAK IS 001230

3K CORE USED
TITLE LPROS

COMMENT

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 17-MAY-74

PART OF LPROS,SHR, VERSION 3

*******************************

SPECIAL NOTE:

LPROS,MAC CONTAINS THE ONLY ENTRY POINT TO THE LPROS HIGH SEGMENT.
THIS ENTRY POINT IS LOCATED IN SECTION B OF THIS PROGRAM.

THE LPROS HIGH SEGMENT IS COMPOSED OF THE FOLLOWING SUBPROGRAMS:

1. LPROS,MAC
2. PROS,MAC
3. CFRAH,MAC
4. AUX4,MAC
5. PRSLOW,MAC (DUMMY LOW SEGMENT)

THESE PROGRAMS SHOULD BE LOADED IN THE ORDER IN WHICH THEY ARE LISTED
ABOVE.  AFTER LOADING, A README BINARY CORE IMAGE OF THE LPROS
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SSAVE MONITOR COMMAND.

THE LPROS HIGH SEGMENT EXECUTES THE FOLLOWING LEARN SYSTEM COMMANDS:

1. PROCESS
2. TEST
3. EP EDIT COMMANDS
4. ET EDIT COMMANDS

*******************************

WHEN LPROS,MAC IS ENTERED FROM ANOTHER HIGH SEGMENT, IT RETRIEVES A
COMMAND INDEX FROM LOCATION STORE AND CALLS THE PROPER SUBPROGRAM TO
EXECUTE THE COMMAND INDICATED BY THE COMMAND INDEX.  AFTER THE COMMAND
HAS BEEN EXECUTED, LPROS,MAC PLACES A COMMAND INDEX IN LOCATION STORE
AND REPLACES ITSELF WITH THE LEARN HIGH SEGMENT (IN THE CASE OF A
TEST COMMAND) OR WITH THE EDIT HIGH SEGMENT (IN THE CASE OF A PROCESS
COMMAND).

CONTENTS OF LPROS,MAC

SECTION A.  PERFORMS THE GETSEG UDO,

SECTION B.  CONTAINS THE ENTRY POINT TO THE LPROS HIGH SEGMENT,
SECTION C. CALLS A SUBPROGRAM TO EXECUTE A "PROCESS" COMMAND.

SECTION D. CALLS A SUBPROGRAM TO EXECUTE AN "EP" EDIT COMMAND.

SECTION E. SETS UP GETSEG ARGUMENTS FOR THE LEDIT HIGH SEGMENT.

SECTION F. CALLS A SUBPROGRAM TO EXECUTE A "TEST" COMMAND.

SECTION G. CALLS A SUBPROGRAM TO EXECUTE AN "ET" EDIT COMMAND.

SECTION H. SETS UP GETSEG ARGUMENTS FOR THE LEARN HIGH SEGMENT.

ENTRY POINT TO LPROS.MAC

RETURN - FROM ANOTHER HIGH SEGMENT, LOCATED IN SECTION B

EXTERNAL SUBROUTINES CALLED BY LPROS.MAC

PROS - EXECUTES "PROCESS" COMMANDS AND "EP" EDIT COMMANDS,
       LOCATED IN PROS.MAC (SECTION A)

PROS2 - EXECUTES "TEST" COMMANDS AND "ET" EDIT COMMANDS, LOCATED
       IN PROS.MAC (SECTION A)

EXTERNAL MEMORY REFERENCES

ARGMT - ARGUMENTS FOR GETSEG UUD (6 WORDS)

STORE - COMMAND INDEX IS SAVED HERE DURING A GETSEG

DESCRIPTION OF FLAGS STORED IN AC=0

BIT 34 TEST COMMAND
BIT 35 COMMAND ERROR

#002000' HISEG

EXTERN ARGMT, PROS, PROS2, STORE

#002000 FLAGS=0
#002001 A=1
#002002 B=2
#002017 Q=17

SECTION A. GET THE PROPER HIGH SEGMENT TO CONTINUE EXECUTION OF
THE COMMAND WHICH WAS TYPED BY THE USER. EXECUTION RESUMES IN
ILEDIT.MAC OR LEARN.MAC.

NOTE: NO DATA OR INSTRUCTION SHOULD BE PLACED AHEAD OF 
"GETSG1", SINCE THE RETURN ADDRESS OF THE GETSEG UGO MUST BE 
THE SAME IN ALL HIGH SEGMENTS.

GETSG1 GETSEG A;
MALT

SECTION B. RETRIEVE THE COMMAND INDEX WHICH WAS PLACED IN LOCATION 
INSTORE* BY THE PREVIOUS HIGH SEGMENT AND JUMP TO THE PROPER ROUTINE 
TO EXECUTE THE COMMAND.

RETURN: MOVE B,STORE; IGET THE COMMAND INDEX IN AC-9

SECTION C. JUMP TO PRCS TO PROCESS THE LESSON.

PRCSS: TRZ FLAGS,2; ICLEAR TEST COMMAND FLAG

SECTION D. JUMP TO PRCS2 TO PROCESS THE LESSON.

PRCSS: LDA 0,PRCS2

SECTION E. SET UP COMMAND INDEX AND GETSEG ARGUMENTS BEFORE.

GETTING THE LEDIT HIGH SEGMENT.

SECTION F. JUMP TO PRCS TO PROCESS THE LESSON.
SECTION G. JUMP TO PROC2 TO PROCESS THE LESSON.

SECTION H. SET UP COMMAND INDEX AND GETSEG ARGUMENTS BEFORE GETTING THE LEARN HIGH SEGMENT.

NO ERRORS DETECTED

PROGRAM BREAK IS 000242

3K CORE USED
TITLE PROS

COMMENT %

WRITTEN BY GARRET A. VAN DER LUGT COMPLETED 18-MAY-74

PART OF LPROS.SHR., VERSION 3

PROS.MAC IS THE MAIN PROGRAM IN THE LEARN LESSON PROCESSOR. THE
LESSON PROCESSOR MAKES THREE PASSES THROUGH THE LESSON. DURING PASS
ONE, KEYWORDS IN EACH LIL STATEMENT ARE CONVERTED TO SINGLE CHARACTER
CODES, AND ERROR CHECKING IS DONE. A TEMPORARY, PARTIALLY PROCESSED
LESSON FILE IS CREATED DURING THIS PASS. THE PARTIALLY PROCESSED
FILE IS SCANNED DURING PASS TWO IN ORDER TO COUNT THE CHARACTERS THAT
PRECED EACH FRAME IN THE LESSON. NO OUTPUT FILE IS CREATED DURING
PASS TWO. DURING PASS THREE, THE FRAME NUMBERS IN ALL BRANCHING
STATEMENTS ARE REPLACED BY THE CHARACTERS COUNTS OBTAINED DURING
PASS TWO. THE FULLY PROCESSED LESSON FILE IS CREATED DURING THE
THIRD PASS.

CONTENTS OF PROS.MAC

SECTION A. INPUTS LESSON NAME AND DOES LESSON LOOKUPS.

SECTION B. INITIALIZES THE LESSON PROCESSOR.

SECTION C. PROCESSES THE FIRST LINE IN A FRAME.

SECTION D. PROCESSES THE FIRST WORD IN A Q-FRAME STATEMENT.

SECTION E. PROCESSES TEXT STATEMENTS.

SECTION F. PROCESSES PAGE AND SLIDE STATEMENTS.

SECTION G. PROCESSES CALC., HINT, BOTH, STORE, REVIEW, AND
RECORD STATEMENTS.

SECTION H. PROCESSES ANSWER STATEMENTS.

SECTION I. PROCESSES EXTRA, ORDER, AND PHONIC STATEMENTS.

SECTION J. PROCESSES DELIM STATEMENTS.

SECTION K. PROCESSES MATCH STATEMENTS.

SECTION L. PROCESSES ANTICIPATED ANSWERS.

SECTION M. PROCESSES ACTION STATEMENTS.

SECTION N. PROCESSES ACTION ANSWER TAGS.
SECTION O. PROCESSES LI STATEMENTS.

SECTION P. PROCESSES FI AND GI STATEMENTS.

SECTION Q. PROCESSES AI STATEMENTS.

SECTION R. PROCESSES SI STATEMENTS.

SECTION S. PROCESSES RI STATEMENTS.

SECTION T. PROCESSES EI STATEMENTS.

SECTION U. PROCESSES GI STATEMENTS.

SECTION V. PROCESSES BI STATEMENTS.

SECTION W. PROCESSES CI STATEMENTS.

SECTION X. PROCESSES R-FRAMES.

SECTION Y. PROCESSES B-FRAMES.

SECTION Z. CALLS SUBROUTINE TO PROCESS C-FRAMES.

SECTION AA. MAKES SECOND PASS THROUGH LESSON.

SECTION BB. MAKES THIRD PASS THROUGH LESSON.

SECTION CC. CLOSES PROCESSED LESSON FILE AND PRINTS ERROR SUMMARY.

LIST OF CODE CHARACTERS PRODUCED BY PROS.MAC

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;00&gt;</td>
<td>TERMINATES TEXT</td>
</tr>
<tr>
<td>&lt;01&gt;</td>
<td>TEXT</td>
</tr>
<tr>
<td>&lt;02&gt;</td>
<td>SLIDE</td>
</tr>
<tr>
<td>&lt;03&gt;</td>
<td>ANSWER</td>
</tr>
<tr>
<td>&lt;04&gt;</td>
<td>END OF LESSON</td>
</tr>
<tr>
<td>&lt;05&gt;</td>
<td>FI</td>
</tr>
<tr>
<td>&lt;06&gt;</td>
<td>RI</td>
</tr>
<tr>
<td>&lt;07&gt;</td>
<td>BI</td>
</tr>
<tr>
<td>&lt;08&gt;</td>
<td>CI</td>
</tr>
<tr>
<td>&lt;09&gt;</td>
<td>EXTRA</td>
</tr>
<tr>
<td>&lt;10&gt;</td>
<td>ORDER</td>
</tr>
<tr>
<td>&lt;11&gt;</td>
<td>PHONIC</td>
</tr>
<tr>
<td>&lt;12&gt;</td>
<td>PAGE</td>
</tr>
<tr>
<td>&lt;13&gt;</td>
<td>CALC</td>
</tr>
<tr>
<td>&lt;14&gt;</td>
<td>DELIM</td>
</tr>
<tr>
<td>&lt;15&gt;</td>
<td>ACTION</td>
</tr>
<tr>
<td>&lt;16&gt;</td>
<td>REVIEW AND MATCH</td>
</tr>
<tr>
<td>&lt;17&gt;</td>
<td>RECORD</td>
</tr>
<tr>
<td>&lt;18&gt;</td>
<td>HINT</td>
</tr>
</tbody>
</table>
ENTRY POINTS TO PROS, MAC

PRCS = FROM LPROS, MAC (SECTIONS C & F), LOCATED IN SECTION A

PRCS2 = FROM LPROS, MAC (SECTIONS D & G), LOCATED IN SECTION A

EXTERNAL SUBROUTINES CALLED BY PROS, MAC

CEXP = PROCESSES ALGEBRAIC EXPRESSION IN AN ANSWER, LOCATED IN CFRAM, MAC (SECTION A)

CFRAM = PROCESSES C-FRAMES, LOCATED IN CFRAM, MAC (SECTION A)

CSTATE = PROCESSES C-FRAME STATEMENT FOLLOWING A CI, LOCATED IN CFRAM, MAC (SECTION A)

FRMLIN = TYPES FRAME AND LINE NOS. IN AN ERROR MESSAGE AND SCANS TO END OF LINE, LOCATED IN AUX4, MAC (SECTION F)

FRMLIN2 = TYPES FRAME AND LINE NOS. IN AN ERROR MESSAGE BUT DOES NOT SCAN TO END OF LINE, LOCATED IN AUX4, MAC (SECTION F)

II = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4, MAC (SECTION H)

I12 = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4, MAC (SECTION H)

I17 = INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B, LOCATED IN AUX4, MAC (SECTION K)

INCHR = INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN AUX4, MAC (SECTION A)

IOD = OUTPUTS AN INTEGER IN AC-A TO THE DISK, LOCATED IN AUX4, MAC (SECTION 0)
10T - OUTPUTS AN INTEGER IN AC-A TO A USER'S TERMINAL, LOCATED IN AUX4.MAC (SECTION E)

OUTCHR - OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION B)

TXTYP - PROCESSES TEXT VARIABLES, LOCATED IN CFRAH.MAC (SECTION A)

TYPSYM - TYPES THE SYMBOL IN AC-G, LOCATED IN AUX4.MAC (SECTION G)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN PRSLOW.MAC):

ADDR - ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO VARIABLES APPEARING IN G-FRAME STATEMENTS

DELS - LIST OF WORD DELIMITERS (17 WORDS)

ERRTOT - TOTAL NO. OF ERRORS DETECTED

FRNM - CURRENT FRAME NO.

FRMTOT - TOTAL NO. OF FRAMES

FRMTYP - CURRENT FRAME TYPE

INAM - DISK LOOKUP ARGUMENTS (4 WORDS)

LNE - CURRENT LINE NO.

NUMLSI - BEGINNING OF CHARACTER COUNT LIST

ONAM - DISK ENTER ARGUMENTS (4 WORDS)

PRT - G-FRAME PART NO.

PC1 - SUBROUTINES STORE RETURN ADDRESS HERE

PC2 - SUBROUTINES STORE RETURN ADDRESS HERE

PC3 - SUBROUTINES STORE RETURN ADDRESS HERE

PSSN - NO. OF LAST FRAME TO BE PROCESSED

PSSST - NO. OF FIRST FRAME TO BE PROCESSED

RFMNT - NO. OF REVIEW FRAMES IN THE LESSON

RSVNUM - TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE

SYMIDX - INDEX TO SYMBOL TABLE
SYMTAB - SYMBOL TABLE FOR VARIABLE NAMES (96 WORDS)

DESCRIPTION OF FLAGS STORED IN AC=0

BIT 13 PERMANENT
BIT 14 ON
BIT 15 TEMPORARY PHONIC
BIT 16 PHONIC ON
BIT 17 TEMPORARY PHONIC ON
BIT 18 NUMERIC ANSWER
BIT 19 TEST COMMAND
BIT 20 COMMAND ERROR

ENTRY PROC, PROC2

EXTERN ADDR, SYMTAB, SYMIDX, RSNUM, CEXP, CSTATE, NUMST, IIT
EXTERN FRMHL, FRMHT, FRMTO, DELS, FRMNM, II, II2, CFRAM, TYPSYM
EXTERN FRMHL1, FRMHL2, IOT, IOD, PRSFN, TXXTP, FRMNL2
EXTERN INCHR, OUTCHR, INAM, OXAM, PC1, PC2, PC3, RF MCU

FLAGS #0
A=1
B=2
C=3
D=4
E=5
F=6
G=7
H=10
Q=17

SECTION A. INPUT LESSON NAME AND LOOKUP THE UNPROCESSED LESSON FILE.

PRCS: MOVEG Q, PC1  ;SAVE RETURN ADDRESS
THE FLGS: 1  ;CLEAR COMMAND ERROR FLAG
SETM TEST  ;SET NO. OF LAST FRAME TO BE PROCESSED TO ZERO
MOVE B, 0999
MOVEB PRSFN
TICALL 4, A  ;INPUT NEXT CHARACTER IN COMMAND STRING
CAI A, 40  ;IS CHARACTER A SPACE?
JASP 92  ;YES, GO INPUT NEXT CHARACTER
PROS MACRO 47 (2041-7) 08159 2-AUG-74 PAGE 1-7

372 002160 326 07 0 002163 NTDF: JUMP 6, +3
373 002161 602 00 0 002202 TRUE FLAGS, 2
374 002162 254 01 0 002165 JRST @PC1
375 002163 251 11 0 002200 TCALL 11, 6
376 002164 251 11 0 002280 TICALL 3, IASC12
377 LESSON NOT FOUND
378 002164 251 11 0 0022544 /)
379 002165 602 02 0 002201 TRUE FLAGS, 1
380 002166 254 02 1 002162 JRST @PC1
381 002167 251 11 0 002280 TICALL 11, 8
382 002168 251 11 0 002280 TICALL 3, IASC12
383 NO_ROOM ON DISK OR QUOTA EXCEEDED
384 002169 251 03 0 0022541 /)
385 002170 662 03 0 002201 TRUE FLAGS, 1
386 002171 254 03 1 002166 JRST @PC1
387 002172 251 11 0 002280 TICALL 11, 8
388
389
390 SECTION B. INITIALIZE THE LESSON PROCESSOR
391 PROC: SETB F, IERRTOT
392 002173 403 00 0 000000 set error count and character count to zero
393 002174 291 01 1 0 002212 MOVIE A, 12
394 002175 202 21 1 0 002200 MOVEM A, RSVNUM RESERVE 10 WORDS OF CORE FOR STUDENT CALC MODE DATA
395 002176 422 20 0 002200 SETF FRMTOT SET FRAME COUNT TO ZERO
396 002177 621 30 0 000000 TLE FLAGS, 7 CLEAR PHONIC ON, TEMP, PHONIC, AND TEMP, PHONIC ON
397 002178 403 20 0 002200 SETM SYMDX SET SYMBOL TABLE INDEX TO ZERO
398 002179 422 20 0 002200 SETM SYMTAB SET 1ST WORD IN SYMBOL TABLE TO ZERO
399 002180 422 20 0 002200 SETM RMCTN SET REVIEW FRAME COUNT TO ZERO
400 002181 422 20 0 002200
401
402 002182 201 21 0 000000 SET UP LIST OF STANDARD WORD DELIMITERS
403 002183 050 20 1 0 000000 MOVIE A, 40
404 002184 202 21 0 000000 MOVEM A, DEL3s PUT A SPACE IN THE LIST
405 002185 201 31 1 0 000004 MOVEM A, "","\n406 002186 222 21 1 0 000000 MOVEM A, DELS+1
407 002187 201 31 1 0 000000 MOVEM A, "\n408 002188 202 21 1 0 000000 MOVEM A, DELS+2
409 002189 222 21 1 0 000000 MOVEM A, "\n410 00218A 222 21 1 0 000000 MOVEM A, DELS+3
411 00218B 201 31 1 0 000000 MOVEM A, "\n412 00218C 222 21 1 0 000000 MOVEM A, DELS+4
413 00218D 222 21 1 0 000000 MOVEM A, "\n414 00218E 222 21 1 0 000000 MOVEM A, DELS+5
415 00218F 201 31 1 0 000000 MOVEM A, "\n416 002190 222 21 1 0 000000 MOVEM A, DELS+6
417 002191 402 20 0 002207 SETM DELS+7 TERMINATE LIST WITH ZERO
418 002192 02223 201 22 1 0 000000 GFSFMM JSP Q, INCHR INPUT CHARACTER
419 002193 306 01 1 0 002204 CAI N A, 4 END OF LESSON?
420 002194 00225 224 00 0 002234 JRST NOPR YES, GO TO NOPR
421 002195 00226 032 01 1 0 000000 CAI N A, 36 BEGINNING OF FRAME?
422 002196 00227 254 00 0 002223 JRST GFSFMM NO, GO NEXT CHARACTER
423 002197 00228 265 17 0 000000 JSP Q, II INPUT FRAME NO.
SECTION C. PROCESS THE FIRST LINE IN A FRAME.

1. Initialize variables
2. Check frame status
3. Process the first line
4. Set flags
5. Return to main process

SECTION D. PROCESS THE FIRST WORD IN A Q-FRAME STATEMENT.

1. Initialize variables
2. Check frame status
3. Process the first word
4. Set flags
5. Return to main process
531 00359' 254 00 0 0 02072; JRST RCMO I SEE SECTION S
532 00360' 316 07 0 0 02058' CAMG G, CSIXBIT /BY/ 3
533 00361' 254 00 0 0 02072; JRST RCMO I SEE SECTION V
534 00362' 316 27 0 0 02062' CAMG G, CSIXBIT /E/ 3
535 00363' 254 02 0 0 02077; JRST ECMD I SEE SECTION T
536 00364' 316 07 0 0 02063' CAMG G, CSIXBIT /C1/ 3
537 00365' 254 00 0 0 02072; JRST RCMO I SEE SECTION W
538 00366' 316 07 0 0 02064' CAMG G, CSIXBIT /G/ 3
539 00367' 254 30 0 0 020214; JRST GCMO I SEE SECTION U
540 00368' 316 27 0 0 02065' CAMG G, CSIXBIT /LI/ 3
541 00369' 254 20 0 0 020225; JRST LCMD I SEE SECTION O
542 00370' 316 27 0 0 02064' CAMG G, CSIXBIT /PI/ 3
543 00371' 254 00 0 0 02072; JRST POCMD I SEE SECTION R
544 00372' 316 07 0 0 02067' CAMG G, CSIXBIT /EXTRA/ 3
545 00373' 254 00 0 0 02072; JRST EXTR I SEE SECTION I
546 00374' 316 27 0 0 02061' CAMG G, CSIXBIT /ORDER/ 3
547 00375' 254 00 0 0 02072; JRST ORDER I SEE SECTION I
548 00376' 316 27 0 0 02061' CAMG G, CSIXBIT /BOTH/ 3
549 00377' 254 00 0 0 02065' JRST BOTH I SEE SECTION G
550 00378' 316 07 0 0 020612; CAMG G, CSIXBIT /HINT/ 3
551 00379' 254 00 0 0 020106; JRST MINT I SEE SECTION G
552 00380' 316 77 0 0 020613; CAMG G, CSIXBIT /CALC/ 3
553 00381' 254 23 0 0 021047; JRST CALC I SEE SECTION G
554 00382' 316 27 0 0 020614; CAMG G, CSIXBIT /PHONIC/ 3
555 00383' 254 23 0 0 0200313; JRST PONIC I SEE SECTION G
556 00384' 316 27 0 0 020615; CAMG G, CSIXBIT /STORE/ 3
557 00385' 254 20 0 0 020104; JRST STORE I SEE SECTION G
558 00386' 316 27 0 0 020616; CAMG G, CSIXBIT /REVIEW/ 3
559 00387' 254 20 0 0 0201101; JRST REVIW I SEE SECTION G
560 00388' 316 27 0 0 020617; CAMG G, CSIXBIT /RECORD/ 3
561 00389' 254 20 0 0 0201112; JRST RECORD I SEE SECTION G
562 00390' 316 07 0 0 0206281; CAMG G, CSIXBIT /REM/ 3
563 00391' 254 20 0 0 0201113; JRST REMARK I SEE SECTION G
564 00392' 316 07 0 0 0206211; CAMG G, CSIXBIT /MATCH/ 3
565 00393' 254 22 0 0 0201414; JRST MATCH I SEE SECTION K
566 00394' 316 27 0 0 0206221; CAMG G, CSIXBIT /DELIM/ 3
567 00395' 254 02 0 0 0201323; JRST DELIM I SEE SECTION J
568 00396' 316 42 0 0 020307; MOVX RO 02 020303; SETS C.
569 00397' 408 03 0 0 020303; LSHB B+36 I SHIFT RIGHT 5 CHARACTERS
570 00398' 266 02 0 0 777742; ADDI B, 40 I CONVERT REMAINING CHARACTER BACK TO SEVEN BIT
571 00399' 267 22 0 0 020240; JUMP G, 1SYM I JUMP IF THERE WERE OTHER CHARACTERS IN AC-A
572 00400' 266 03 0 0 277481; JRST ATAC I SEE SECTION N
573 00401' 264 03 0 0 020754; REMARK: JSP Q, INCHR I INPUT CHARACTER
574 00402' 265 17 0 0 0200000; OSYMER: JSP Q, FRMLIN I TYPE FRAME AND LINE NO.
575 00403' 265 17 0 0 0205321; TDCALL 3, SACODE /}
576 00404' 265 17 0 0 0205230; ILLEGAL G, FRAME STATEMENT /
577 00405' 265 17 0 0 0205230; JSP Q, TPSYM I TYPE ILLEGAL SYMBOL
578 00406' 265 00 0 0 020276; JRST QFRAME
579 00407' 265 17 0 0 0200262; ENOLIN: JSP Q, OUTCHR I OUTPUT THE RETURN CHARACTER
580 00408' 265 17 0 0 020317; JSP Q, INCHR I INPUT LINE FEED
581 00409' 265 20 0 0 020276; JRST QFRAME
582 00410' 265 17 0 0 0205445; REMARK: JSP Q, INCHR I INPUT CHARACTER
583 00411' 269 17 0 0 0205445;
### SECTION E. PROCESS A TEXT STATEMENT

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>584</td>
<td>0024551</td>
<td>JST, A15</td>
</tr>
<tr>
<td>585</td>
<td>0024531</td>
<td>JST, B1</td>
</tr>
<tr>
<td>586</td>
<td>0024521</td>
<td>JST, G1</td>
</tr>
<tr>
<td>587</td>
<td>0024541</td>
<td>JST, INCHR</td>
</tr>
<tr>
<td>588</td>
<td>0024551</td>
<td>TEXTT, MOVE B, PART</td>
</tr>
<tr>
<td>589</td>
<td>0024561</td>
<td>CAIE B, 1</td>
</tr>
<tr>
<td>590</td>
<td>0024571</td>
<td>CAIE B, 17</td>
</tr>
<tr>
<td>591</td>
<td>0024531</td>
<td>QERR1, MOV. B, A</td>
</tr>
<tr>
<td>592</td>
<td>0024561</td>
<td>QERR1, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>593</td>
<td>0024571</td>
<td>QERR1, SAVE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>594</td>
<td>0024551</td>
<td>RECOV1, MOVE A, 1</td>
</tr>
<tr>
<td>595</td>
<td>0024531</td>
<td>RECOV1, SET PART NO. TO 1</td>
</tr>
<tr>
<td>596</td>
<td>0024561</td>
<td>RECOV1, OUTPUT (210)</td>
</tr>
<tr>
<td>597</td>
<td>0024571</td>
<td>RECOV1, OUTPUT THE RETURN CHARACTER</td>
</tr>
<tr>
<td>598</td>
<td>0024551</td>
<td>TEXT2I, CAIE A, &quot;S&quot;</td>
</tr>
<tr>
<td>599</td>
<td>0024561</td>
<td>TEXT2I, JST TT</td>
</tr>
<tr>
<td>600</td>
<td>0024571</td>
<td>TEXT2I, IYES, GO TO TT</td>
</tr>
<tr>
<td>601</td>
<td>0024551</td>
<td>TEXT2I, CAIE A, RETURN</td>
</tr>
<tr>
<td>602</td>
<td>0024561</td>
<td>TEXT2I, IN, JUMP</td>
</tr>
<tr>
<td>603</td>
<td>0024571</td>
<td>TEXT2I, OUTPUT THE RETURN CHARACTER</td>
</tr>
<tr>
<td>604</td>
<td>0024551</td>
<td>ACTXTI, JSP Q, OUTCHR</td>
</tr>
<tr>
<td>605</td>
<td>0024561</td>
<td>ACTXTI, IPRINT CHARACTER</td>
</tr>
<tr>
<td>606</td>
<td>0024571</td>
<td>ACTXTI, QERR1, OUTPUT CHARACTER</td>
</tr>
<tr>
<td>607</td>
<td>0024551</td>
<td>TT1I, JSP Q, INCHR</td>
</tr>
<tr>
<td>608</td>
<td>0024561</td>
<td>TT1I, IPRINT CHARACTER</td>
</tr>
<tr>
<td>609</td>
<td>0024571</td>
<td>TT1I, QERR1, OUTPUT CHARACTER</td>
</tr>
<tr>
<td>610</td>
<td>0024551</td>
<td>TT7I, JSP Q, INCHR</td>
</tr>
<tr>
<td>611</td>
<td>0024561</td>
<td>TT7I, IPRINT CHARACTER</td>
</tr>
<tr>
<td>612</td>
<td>0024571</td>
<td>TT7I, QERR1, OUTPUT CHARACTER</td>
</tr>
<tr>
<td>613</td>
<td>0024551</td>
<td>CAIN A, &quot;&quot;</td>
</tr>
<tr>
<td>614</td>
<td>0024561</td>
<td>CAIN A, JST SLASH</td>
</tr>
<tr>
<td>615</td>
<td>0024571</td>
<td>CAIN A, IYES, GO TO SLASH</td>
</tr>
<tr>
<td>616</td>
<td>0024551</td>
<td>CAIN A, &quot;A&quot;</td>
</tr>
<tr>
<td>617</td>
<td>0024561</td>
<td>CAIN A, IEND OF LESSON</td>
</tr>
<tr>
<td>618</td>
<td>0024571</td>
<td>CAIN A, IEND OF FRAME</td>
</tr>
<tr>
<td>619</td>
<td>0024551</td>
<td>CAIN A, &quot;S&quot;</td>
</tr>
<tr>
<td>620</td>
<td>0024561</td>
<td>CAIN A, IYES, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>621</td>
<td>0024571</td>
<td>CAIN A, IYES, GO TO LINE FEED</td>
</tr>
<tr>
<td>622</td>
<td>0024551</td>
<td>CAIN A, IYES, INCREMENT LINE COUNT</td>
</tr>
<tr>
<td>623</td>
<td>0024561</td>
<td>CAIN A, IGO OUTPUT CHARACTER</td>
</tr>
<tr>
<td>624</td>
<td>0024571</td>
<td>CAIN A, ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>625</td>
<td>0024551</td>
<td>SQERR1, JSP Q, FRMLN2</td>
</tr>
<tr>
<td>626</td>
<td>0024561</td>
<td>SQERR1, ISAVE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>627</td>
<td>0024571</td>
<td>SQERR1, ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>628</td>
<td>0024551</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>629</td>
<td>0024561</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>630</td>
<td>0024571</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>631</td>
<td>0024551</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>632</td>
<td>0024561</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>633</td>
<td>0024571</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>634</td>
<td>0024551</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>635</td>
<td>0024561</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>636</td>
<td>0024571</td>
<td>TTCELL 3, IASCIZ /</td>
</tr>
<tr>
<td>PROS</td>
<td>MACRO</td>
<td>47(284)+7 00190 2-AUG-75 PAGE 1-14</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>743</td>
<td>0026701</td>
<td>254 00 00 00 007118 JRST PGERR1; NO, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>744</td>
<td>0026712</td>
<td>202 00 00 00 006799 MOVEM B.PART; SET PART NO. TO 1</td>
</tr>
<tr>
<td>745</td>
<td>0026713</td>
<td>200 00 00 00 006801 MOVE B,A; SAVE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>746</td>
<td>0026714</td>
<td>201 00 00 00 006802 MOVEI A,16</td>
</tr>
<tr>
<td>747</td>
<td>0026715</td>
<td>265 17 00 00 006803 MOVE A,B; OUTPUT &lt;15&gt;</td>
</tr>
<tr>
<td>748</td>
<td>0026716</td>
<td>200 01 00 00 006804 MOVE A,B; PUT CHARACTER BACK IN AC-A</td>
</tr>
<tr>
<td>749</td>
<td>0026717</td>
<td>254 00 00 00 006805 JRST +2</td>
</tr>
<tr>
<td>750</td>
<td>0026718</td>
<td>265 17 00 00 006806 JSPG Q.INCHR; INPUT CHARACTER</td>
</tr>
<tr>
<td>751</td>
<td>0026719</td>
<td>302 01 00 00 006807 CAIE A,11</td>
</tr>
<tr>
<td>752</td>
<td>0026720</td>
<td>302 01 00 00 006808 CAIE A,48</td>
</tr>
<tr>
<td>753</td>
<td>0026721</td>
<td>254 20 00 00 006809 JRST -3</td>
</tr>
<tr>
<td>754</td>
<td>0026722</td>
<td>265 17 00 00 006809 JRST PGERR1; INPUT CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>755</td>
<td>0026723</td>
<td>302 21 00 00 006809 JRST PGERR2; INPUT CHARACTER IS A RETURN</td>
</tr>
<tr>
<td>756</td>
<td>0026724</td>
<td>265 17 00 00 006809 JRST PGERR1; IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>757</td>
<td>0026725</td>
<td>265 17 00 00 006809 JRST PGERR1; IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>758</td>
<td>0026726</td>
<td>265 17 00 00 006809 JRST OFRAME; GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>759</td>
<td>0026727</td>
<td>265 17 00 00 006809 PGERR1; JSP Q.FRMLIN; TYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>760</td>
<td>0026728</td>
<td>265 17 00 00 006809 ITCALL 3,[ASCZ]</td>
</tr>
<tr>
<td>761</td>
<td>0026729</td>
<td>251 33 00 00 002662 PAGE STATEMENT AFTER ANSWER OR ACTION STATEMENT/3</td>
</tr>
<tr>
<td>762</td>
<td>0026730</td>
<td>254 00 00 00 002674 JRST OFRAME; GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>763</td>
<td>0026731</td>
<td>265 17 00 00 002711 JRST OFRAME; GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>764</td>
<td>0026732</td>
<td>265 17 00 00 002674 ILLEGAL PAGE STATEMENT FORMAT/3</td>
</tr>
<tr>
<td>765</td>
<td>0026733</td>
<td>254 00 00 00 002711 SLIDE1</td>
</tr>
<tr>
<td>766</td>
<td>0026734</td>
<td>265 20 00 00 002674 MOVEM A,1</td>
</tr>
<tr>
<td>767</td>
<td>0026735</td>
<td>281 21 00 00 002674 JRST SGERR2; EXIT TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>768</td>
<td>0026736</td>
<td>254 20 00 00 002674 CAIE A,15</td>
</tr>
<tr>
<td>769</td>
<td>0026737</td>
<td>254 00 00 00 002674 CAIE A,101</td>
</tr>
<tr>
<td>770</td>
<td>0026738</td>
<td>202 01 00 00 002674 JRST SGERR2; EXIT TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>771</td>
<td>0026739</td>
<td>265 21 00 00 002674 MOVEM A,1; SET PART NO. TO 1</td>
</tr>
<tr>
<td>772</td>
<td>0026740</td>
<td>265 17 00 00 002674 RECQV1</td>
</tr>
<tr>
<td>773</td>
<td>0026741</td>
<td>265 17 00 00 002674 JRST OFRAME; GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>774</td>
<td>0026742</td>
<td>265 17 00 00 002674 JSP Q.INCHR; INPUT CHARACTER</td>
</tr>
<tr>
<td>775</td>
<td>0026743</td>
<td>265 17 00 00 002674 CAIE A,11</td>
</tr>
<tr>
<td>776</td>
<td>0026744</td>
<td>265 17 00 00 002674 CAIE A,48</td>
</tr>
<tr>
<td>777</td>
<td>0026745</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>778</td>
<td>0026746</td>
<td>302 01 00 00 002674 CAIE A,15</td>
</tr>
<tr>
<td>779</td>
<td>0026747</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>780</td>
<td>0026748</td>
<td>302 01 00 00 002674 CAIE A,101</td>
</tr>
<tr>
<td>781</td>
<td>0026749</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>782</td>
<td>0026750</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>783</td>
<td>0026751</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>784</td>
<td>0026752</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>785</td>
<td>0026753</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>786</td>
<td>0026754</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>787</td>
<td>0026755</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>788</td>
<td>0026756</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>789</td>
<td>0026757</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>790</td>
<td>0026758</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>791</td>
<td>0026759</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>792</td>
<td>0026760</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>793</td>
<td>0026761</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>794</td>
<td>0026762</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>795</td>
<td>0026763</td>
<td>302 01 00 00 002674 JRST ERR7; IF END OF STATEMENT, GO TYPE ERROR MESSAGE</td>
</tr>
</tbody>
</table>

---
<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>649</td>
<td><em>01837</em> 251 03 0 00 002761</td>
</tr>
<tr>
<td>650</td>
<td><em>01840</em> 254 02 0 00 002276</td>
</tr>
<tr>
<td>651</td>
<td><em>01841</em> 265 17 0 00 001035</td>
</tr>
<tr>
<td>652</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>653</td>
<td><em>01842</em> 251 03 0 00 002725</td>
</tr>
<tr>
<td>654</td>
<td><em>01843</em> 254 02 0 00 002276</td>
</tr>
<tr>
<td>655</td>
<td><em>01844</em> 265 17 0 00 001035</td>
</tr>
<tr>
<td>656</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>657</td>
<td><em>01845</em> 251 03 0 00 002736</td>
</tr>
<tr>
<td>658</td>
<td><em>01846</em> 254 02 0 00 002761</td>
</tr>
</tbody>
</table>

| SECTION 6. PROCES A CALC, HINT, BOTH, STORE, REVIEW, OR RECORD |
| STATEMENT: |

<table>
<thead>
<tr>
<th>Line</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>661</td>
<td>MOVEI B,2</td>
</tr>
<tr>
<td>662</td>
<td>CALC:</td>
</tr>
<tr>
<td>663</td>
<td>MOVEI B,2</td>
</tr>
<tr>
<td>664</td>
<td>CAMLE B, PRT</td>
</tr>
<tr>
<td>665</td>
<td>JST CALCOK</td>
</tr>
<tr>
<td>666</td>
<td>JSP 0,FRLN2</td>
</tr>
<tr>
<td>667</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>668</td>
<td>CALC CHANGE DOES NOT PRECEDE ANSWER STATEMENT/</td>
</tr>
<tr>
<td>669</td>
<td>CALCOK: MOVEI A,17</td>
</tr>
<tr>
<td>670</td>
<td>JST ONOFF</td>
</tr>
<tr>
<td>671</td>
<td>HINT:</td>
</tr>
<tr>
<td>672</td>
<td>CAMLE B, PRT</td>
</tr>
<tr>
<td>673</td>
<td>JST HINTOK</td>
</tr>
<tr>
<td>674</td>
<td>JSP 0,FRLN2</td>
</tr>
<tr>
<td>675</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>676</td>
<td>HINT STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/</td>
</tr>
<tr>
<td>677</td>
<td>HINTOK: MOVEI A,26</td>
</tr>
<tr>
<td>678</td>
<td>BOTH:</td>
</tr>
<tr>
<td>679</td>
<td>CAMLE B, PRT</td>
</tr>
<tr>
<td>680</td>
<td>JST BOTHOK</td>
</tr>
<tr>
<td>681</td>
<td>JSP 0,FRLN2</td>
</tr>
<tr>
<td>682</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>683</td>
<td>BOTH STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/</td>
</tr>
<tr>
<td>684</td>
<td>BOTHOK: MOVEI A,27</td>
</tr>
<tr>
<td>685</td>
<td>JST ONOFF</td>
</tr>
<tr>
<td>686</td>
<td>STOREI MOVEI B,2</td>
</tr>
<tr>
<td>687</td>
<td>CAMLE B, PRT</td>
</tr>
<tr>
<td>688</td>
<td>JST STOREOK</td>
</tr>
<tr>
<td>689</td>
<td>JSP 0,FRLN2</td>
</tr>
<tr>
<td>690</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
<tr>
<td>691</td>
<td>STORE STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/</td>
</tr>
<tr>
<td>692</td>
<td>STOREOK: MOVEI A,34</td>
</tr>
<tr>
<td>693</td>
<td>JST ONOFF</td>
</tr>
<tr>
<td>694</td>
<td>REVIEW: MOVEI B,2</td>
</tr>
<tr>
<td>695</td>
<td>CAMLE B, PRT</td>
</tr>
<tr>
<td>696</td>
<td>JST REVOK</td>
</tr>
<tr>
<td>697</td>
<td>JSP 0,FRLN2</td>
</tr>
<tr>
<td>698</td>
<td>TCCALL 3, [ASCII /</td>
</tr>
</tbody>
</table>
SECTION 1. PROCESS EXTRA, ORDER, AND PHONIC STATEMENTS.

1032 001275 ' 201 02 00 000202  EXTRA: MOVEI 9, 2
1033 001276 ' 316 02 00 001217*  CALL B, PART 1 HAS ANSWER STATEMENT BEEN ENCOUNTERED?
1034 001277 ' 254 00 00 001302*  JRST EXTOK 1 YES, GO TO EXTOK
1035 001278 ' 265 17 00 001264*  JRST FRMLN2  TYPE FRAME AND LINE NOS.
1036 001279  TCALL 3, [ASCII]
1037 001280 ' 251 03 00 0023107*  EXTRA STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/
1038 001281 ' 201 01 00 002011  EXTOK 1 MOVEI 4, 11  INPUT STATEMENT CODE IN AC-A
1039 001282 ' 254 02 00 001120  JRST QONFF 1 SEE SECTION G
1040 001283 ' 316 02 00 002326*  ORDER: MOVEI 9, 2
1041 001284 ' 201 02 00 002202*  JRST QONFF 1 SEE SECTION G
1042 001285 ' 316 02 00 002326*  ORDER: MOVEI 9, 2
1043 001286 ' 254 02 00 001311*  JRST QONFF 1 SEE SECTION G
1044 001287 ' 265 17 00 001320*  JRST QONFF 1 SEE SECTION G
1045 001288  TCALL 3, [ASCII]
1046 001289 ' 251 03 00 003122*  ORDER STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/
1047 001290 ' 201 01 00 003213  ORDDOK 1 MOVEI 4, 13  INPUT STATEMENT CODE IN AC-A
1048 001291 ' 220 02 00 001129*  JRST QONFF 1 SEE SECTION G
1049 001292 ' 316 02 00 003335*  ORDER: MOVEI 9, 2
1050 001293 ' 201 02 00 002202*  JRST QONFF 1 SEE SECTION G
1051 001294 ' 220 02 00 003335  JRST PHONOK 1 YES, GO TO PHONOK
1052 001295 ' 201 02 00 003335  JRST PHONOK 1 YES, GO TO PHONOK
1053 001296 ' 220 02 00 003335  JRST PHONOK 1 YES, GO TO PHONOK
1054 001297 ' 265 17 00 003837*  JRST QONFF 1 SEE SECTION G
1055 001298  TCALL 3, [ASCII]
1056 001299 ' 251 03 00 003135*  PHONIC STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/
1057 001300 ' 474 04 00 000000  SET AC-D TO ONES TO INDICATE PHONIC
1058 001301 ' 201 01 00 000914  MOVEI 4, 14  INPUT STATEMENT CODE IN AC-A
1059 001302 ' 220 02 00 001129*  JRST QONFF 1 SEE SECTION G
1060 001303  END
SECTION L. PROCESS ANTICIPATED ANSWERS.

1167    021451* 254 02 00 000002  JSP Q,OUTCHR :PUT DELIMITER IN AC-A
1168    021451* 254 02 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1169    021473* 306 01 00 000095  MOVE A,B  :INPUT DELIMITER IN AC-A
1170    021473* 306 01 00 000095  MOVE A,B  :INPUT DELIMITER IN AC-A
1171    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1172    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1173    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1174    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1175    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1176    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1177    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1178    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1179    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1180    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1181    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1182    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1183    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1184    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1185    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1186    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1187    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1188    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1189    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1190    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1191    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1192    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1193    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1194    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1195    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1196    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1197    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1198    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1199    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1200    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1201    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1202    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1203    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1204    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1205    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1206    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1207    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1208    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1209    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1210    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1211    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1212    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1213    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1214    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1215    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1216    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1217    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1218    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
1219    021449* 250 01 00 000002  JSP Q,OUTCHR :INPUT NEXT CHARACTER
SECTION Y. PROCESS A B-FRAME.

1612 002237 265 17 0 00 002230 BFRAME: JSP G. INCHR 1 INPUT CHARACTER
1613 002240 336 01 0 00 003036 CAIN A.36 1 BEGINNING OF NEXT FRAME?
1614 002241 254 00 0 00 002401 JRSF FSTLIN 1 YES, GO TO FSTLIN (SECTION C)
1615 002242 336 01 0 00 003036 CAIN A.4 1 END OF LESSON?
1616 002243 254 00 0 00 002251 JRSF EOL1 1 YES, GO TO EOL1 (SECTION AA)
1617 002244 254 17 0 00 002253 JSP G. OUTCHR 1 ELSE, OUTPUT CHARACTER
1618 002245 254 00 0 00 002237 JRSF BFRAME 1 GO INPUT NEXT CHARACTER

SECTION Z. PROCESS A C-FRAME.

1624 002246 265 17 0 00 002000 CFRAIIE: JSP G. CFRAIIE 1 GO PROCESS THE C-FRAME
1625 002247 254 00 0 00 002401 JRSF FSTLIN 1 IF BEGINNING OF NEXT FRAME, GO TO FSTLIN (SECTION C)
1626 002248 254 00 0 00 002251 JRSF EOL1 1 IF END OF LESSON, GO TO EOL1 (SECTION AA)

SECTION AA. MAKE A SECOND PASS THRU THE LESSON TO COUNT THE
NUMBER OF CHARACTERS PRECEDING EACH FRAME.

1632 002251 201 01 0 00 002004 EDL1: MOVEM A.4
1633 002252 265 17 0 00 002444 JSP G. OUTCHR 1 OUTPUT <B4> TO INDICATE END OF LESSON
1634 002253 270 01 0 00 002000 CLOSE 1 1 CLOSE UNPROCESSED LESSON FILE
1635 002255 270 02 0 00 002000 CLOSE 2 1 CLOSE PARTIALLY PROCESSED LESSON FILE
1636 002256 205 01 0 00 045560 MOVE A.1'THP'
1637 002257 205 01 0 00 003001 MOVE A.INAM+1 1 MOVE EXTENSION TO LOOKUP ARGUMENTS
1638 002258 205 01 0 00 044263 MOVE A.'PRS'
1639 002259 202 01 0 00 003001 MOVE A.ONAM+1 1 MOVE EXTENSION TO ENTER ARGUMENTS
1640 002260 482 00 0 00 003002 SETAM INAM+2
1641 002261 482 00 0 00 003002 SETAM ONAM+2
1642 002262 482 00 0 00 003003 SETAM INAM+3
1643 002263 482 00 0 00 003003 SETAM ONAM+3
<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1823</td>
<td>022521</td>
<td>351</td>
<td>03</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1824</td>
<td>022522</td>
<td>200</td>
<td>01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1825</td>
<td>022523</td>
<td>201</td>
<td>01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1826</td>
<td>022524</td>
<td>366</td>
<td>00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1827</td>
<td>022525</td>
<td>254</td>
<td>20</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1828</td>
<td>022526</td>
<td>402</td>
<td>02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1829</td>
<td>022527</td>
<td>402</td>
<td>02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1830</td>
<td>022510</td>
<td>055</td>
<td>02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1831</td>
<td>022511</td>
<td>254</td>
<td>04</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1832</td>
<td>022512</td>
<td>254</td>
<td>00</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 003325

5K CORE USED
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
<th>Value 4</th>
<th>Value 5</th>
<th>Value 6</th>
<th>Value 7</th>
<th>Value 8</th>
<th>Value 9</th>
<th>Value 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCMOD</td>
<td>633</td>
<td>1923#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCMOD1</td>
<td>1511</td>
<td>1532#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFRAFME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNO</td>
<td>1716</td>
<td>1716</td>
<td>1716#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNODEA</td>
<td>1934</td>
<td>1957#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BTH</td>
<td>639</td>
<td>639#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOTHOK</td>
<td>683</td>
<td>887#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>241#</td>
<td>268</td>
<td>277</td>
<td>281</td>
<td>347</td>
<td>356</td>
<td>358</td>
<td>475</td>
<td>504</td>
<td>509</td>
</tr>
<tr>
<td></td>
<td>714</td>
<td>717</td>
<td>765</td>
<td>791</td>
<td>860</td>
<td>812</td>
<td>1086</td>
<td>1011</td>
<td>1367</td>
<td>1360</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALC</td>
<td>533</td>
<td>865#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALC0K</td>
<td></td>
<td>671#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CEXP</td>
<td>233#</td>
<td>120#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CFRAF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMOD1</td>
<td>1176</td>
<td>1572#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSTATE</td>
<td>233#</td>
<td>1572</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>242#</td>
<td>346</td>
<td>361</td>
<td>912</td>
<td>949</td>
<td>1057</td>
<td>130#</td>
<td>1304</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATPNT</td>
<td>251#</td>
<td>347</td>
<td>395</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELERR</td>
<td>1253</td>
<td>1117#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELIN</td>
<td>567</td>
<td>1065#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELS</td>
<td>234#</td>
<td>424</td>
<td>406</td>
<td>408</td>
<td>410</td>
<td>412</td>
<td>414</td>
<td>416</td>
<td>417</td>
<td>1086</td>
</tr>
<tr>
<td>DELSIN</td>
<td>1988</td>
<td>1291#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLSQER</td>
<td>1267</td>
<td>1121#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>243#</td>
<td>621</td>
<td>625</td>
<td>1768</td>
<td>1772</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECMOD</td>
<td>535</td>
<td>1485#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENLIN</td>
<td>452</td>
<td>580#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOL1</td>
<td></td>
<td>484</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOL2</td>
<td>1636#</td>
<td>1636#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOL3</td>
<td>1716</td>
<td>1716#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR10</td>
<td>1273</td>
<td>1225#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR11</td>
<td>1546</td>
<td>1556#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR12</td>
<td>1534</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR14</td>
<td>1534</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR4</td>
<td>617</td>
<td>627#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR5</td>
<td>725</td>
<td>732#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR7</td>
<td>779</td>
<td>856#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR9</td>
<td>794</td>
<td>651#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRTOT</td>
<td>243#</td>
<td>392</td>
<td>1800</td>
<td>1800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXPRES</td>
<td>1191</td>
<td>1205#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTRR</td>
<td>1037</td>
<td>1041#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCH0</td>
<td>555</td>
<td>1535#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCMD</td>
<td>1428</td>
<td>1418#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCMD3</td>
<td>1428</td>
<td>1418#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLAGS</td>
<td>238#</td>
<td>258</td>
<td>286</td>
<td>292</td>
<td>298</td>
<td>301</td>
<td>373</td>
<td>379</td>
<td>385</td>
<td>396</td>
</tr>
<tr>
<td></td>
<td>932</td>
<td>950#</td>
<td>992</td>
<td>953</td>
<td>954</td>
<td>995</td>
<td>957</td>
<td>958</td>
<td>959</td>
<td>968</td>
</tr>
<tr>
<td>FNUMIN</td>
<td>1712</td>
<td>1767#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECTION 5. PROCESSES RETURN STATEMENTS.

LIST OF CHARACTER CODES PRODUCED BY CFRAH,MAC

<table>
<thead>
<tr>
<th>Character Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;00&gt;</td>
<td>PRECEDES LINE NUMBER</td>
</tr>
<tr>
<td>&lt;02&gt;</td>
<td>PRECEDES REGULAR VARIABLE ADDRESS</td>
</tr>
<tr>
<td>&lt;03&gt;</td>
<td>INPUT</td>
</tr>
<tr>
<td>&lt;04&gt;</td>
<td>ATYPE</td>
</tr>
<tr>
<td>&lt;07&gt;</td>
<td>BI</td>
</tr>
<tr>
<td>&lt;10&gt;</td>
<td>GOTO</td>
</tr>
<tr>
<td>&lt;11&gt;</td>
<td>FILL</td>
</tr>
<tr>
<td>&lt;12&gt;</td>
<td>TYPE</td>
</tr>
<tr>
<td>&lt;13&gt;</td>
<td>PRECEDES OPERATOR CODE</td>
</tr>
<tr>
<td>&lt;14&gt;</td>
<td>PRECEDES CONSTANT</td>
</tr>
<tr>
<td>&lt;20&gt;</td>
<td>PRECEDES SPECIAL VARIABLE CODE</td>
</tr>
<tr>
<td>&lt;21&gt;</td>
<td>PRECEDES ARRAY VARIABLE ADDRESS</td>
</tr>
<tr>
<td>&lt;30&gt;</td>
<td>GI</td>
</tr>
<tr>
<td>&lt;31&gt;</td>
<td>GI (RETURN)</td>
</tr>
<tr>
<td>&quot;C&quot;</td>
<td>CALL</td>
</tr>
<tr>
<td>&quot;R&quot;</td>
<td>RETURN</td>
</tr>
</tbody>
</table>

ENTRY POINTS TO CFRAH,MAC

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEXP</td>
<td>FROM PROS,MAC (SECTION 1), LOCATED IN SECTION A</td>
</tr>
<tr>
<td>CFRAH</td>
<td>FROM PROS,MAC (SECTION 2), LOCATED IN SECTION A</td>
</tr>
<tr>
<td>CSTATE</td>
<td>FROM PROS,MAC (SECTION H), LOCATED IN SECTION A</td>
</tr>
<tr>
<td>TXTYP</td>
<td>FROM PROS,MAC (SECTION E), LOCATED IN SECTION A</td>
</tr>
</tbody>
</table>

EXTERNAL SUBROUTINES CALLED BY CFRAH,MAC

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOUT</td>
<td>OUTPUTS THE FLOATING POINT NO. IN AC-A TO THE DISK, LOCATED IN AUX4,MAC (SECTION J)</td>
</tr>
<tr>
<td>FRMLIN</td>
<td>TYPES FRAME AND LINE NO. IN AN ERROR MESSAGE AND SCANS TO END OF LINE, LOCATED IN AUX4,MAC (SECTION F)</td>
</tr>
<tr>
<td>II</td>
<td>INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4,MAC (SECTION H)</td>
</tr>
<tr>
<td>II2</td>
<td>INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4,MAC (SECTION H)</td>
</tr>
<tr>
<td>INCHR</td>
<td>INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN AUX4,MAC (SECTION A)</td>
</tr>
<tr>
<td>INFLP</td>
<td>INPUTS A FLOATING POINT NUMBER FROM THE DISK INTO AC-B, LOCATED IN AUX4,MAC (SECTION I)</td>
</tr>
</tbody>
</table>
213 083015' 265 17 00 000000*  JSP G.FRMLIN ELSE: TYPE FRAME AND LINE NOS.
214
215 083216' 251 03 00 000021' TT CALL 3, LASC1E /
216 083217* 254 02 00 000024* ANSWER EXPRESSION NOT DELIMITED BY A COMMA/
217 083221* 222 17 00 000017* JRT SP,PC4 RETURN TO PROG, MAC
218 083221* 226 17 00 000024* CS TATEJ MOVER G,PC4 SAVE RETURN ADDRESS
219 083221* 226 17 00 000020* TR,FLAGS,22 SET G-FRAME FLAG
220 083221* 246 02 00 000021* JRT ITER GO TO ITER (SECTION C)
221 083221* 250 02 00 000020* MOVE B,A SAVE CHARACTER IN AC-A
222 083221* 221 17 00 000013* MOVEI A,13 PUT TYPE STATEMENT CODE <13> IN AC-A
223 083221* 246 17 00 000021* JRT PC,PC4 SAVE RETURN ADDRESS
224 083221* 220 01 00 000022* MOVE A,B, PUT CHARACTER BACK IN AC-A
225 083221* 254 20 00 000020* TXTYP, TXTYPE SET
226 083221* 254 20 00 000021* GO TO TXTYP (SECTION M)
227 083221* 222 17 00 000023* MOVER G,PC4 SAVE RETURN ADDRESS
228 083221* 402 02 00 000020* SET, LINENO CLEAR LINE NO., BITS
229 083221* 402 02 00 000020* SET, GOTO NO. CLEAR GOTO NO., BITS
230 083221* 626 02 00 000020* TR,FLAGS,20 CLEAR G-FRAME FLAG
231
232
233
234
235
236 083031* 602 00 00 000020* LINENUM TRNE FLAG,22 IS G-FRAME FLAG SET?
237 083037* 254 02 10 000012* JRT SP,PC4 YES, RETURN
238 083049* 350 02 00 000020* MOVER G,PC4 YES, INCREMENT LINE NUMBER
239 083049* 265 17 00 000020* LPNHI JSP I,INCHR INPUT CHARACTER
240 083049* 322 01 00 000015* CAI, A,11
241 083049* 326 01 00 000011* CAI, A,12
242 083049* 254 02 00 000041* JRT LHM IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
243 083049* 322 01 00 000015* CAI, A,17
244 083049* 326 01 00 000011* CAI, A,18
245 083049* 254 02 00 000041* JRT LHM IF END OF LINE, GO TO LINENUM
246 083049* 326 01 00 000013* CAI, A,36 END OF FRAME?
247 083049* 254 02 00 000041* JRT CKNOS YES, GO CHECK LINE NOS.
248 083049* 326 01 00 000013* CAI, A,4 END OF LESSON?
249 083049* 360 02 00 000037* MOVER G,PC4 YES, INCREMENT RETURN ADDRESS
250 083049* 326 01 00 000030* CAI, A,4 END OF LESSON?
251 083049* 360 02 00 000020* MOVER G,PC4 YES, GO CHECK LINE NOS.
252 083049* 301 01 00 000060* CAI, A,69
253 083057* 301 02 00 000071* CAI, A,71
254 083057* 301 01 00 000071* JRT ITER2 IF CHARACTER IS NOT A NUMBER, GO TO ITER2 (SECTION C)
255 083057* 326 01 00 000013* CAI, A,111
256 083057* 326 01 00 000013* JSP I,INCHR INPUT LINE NO.
257 083057* 326 01 00 000013* JUNPE 8, NUMER1 IF LINE NO. IS ZERO, GO TYPE ERROR MESSAGE
258 083057* 301 22 00 000031* CALL B,31 J' I, LINE NO. LESS THAN 25?
259 083057* 254 02 00 000013* JRT NUMER1 NO, GO TYPE ERROR MESSAGE
260 083057* 305 21 00 000024* CAI, A,40 DOES A SPACE FOLLOW THE NO.?
261 083057* 254 02 00 000024* JRT A,4 YES, JUMP
262 083057* 302 21 00 000020* CAI, A,14 DOES A TAB FOLLOW THE NO.?
263 083057* 254 02 00 000015* JRT NUMER1 NO, GO TYPE ERROR MESSAGE
264 083057* 281 32 00 000001* MOVER D,LINENO PUT LINE NO. BITS IN AC-D
265 083057* 283 34 00 000033*
SECTION C. PROCESSES ITERATION FIELDS

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>301</td>
<td>021351</td>
</tr>
<tr>
<td>302</td>
<td>02126</td>
</tr>
<tr>
<td>303</td>
<td>02127</td>
</tr>
<tr>
<td>304</td>
<td>02130</td>
</tr>
<tr>
<td>305</td>
<td>02131</td>
</tr>
<tr>
<td>306</td>
<td>02132</td>
</tr>
<tr>
<td>307</td>
<td>02133</td>
</tr>
<tr>
<td>308</td>
<td>02134</td>
</tr>
<tr>
<td>309</td>
<td>02135</td>
</tr>
<tr>
<td>310</td>
<td>02136</td>
</tr>
<tr>
<td>311</td>
<td>02137</td>
</tr>
<tr>
<td>312</td>
<td>02138</td>
</tr>
<tr>
<td>313</td>
<td>02139</td>
</tr>
<tr>
<td>314</td>
<td>02140</td>
</tr>
<tr>
<td>315</td>
<td>02141</td>
</tr>
<tr>
<td>316</td>
<td>02142</td>
</tr>
<tr>
<td>317</td>
<td>02143</td>
</tr>
<tr>
<td>318</td>
<td>02144</td>
</tr>
</tbody>
</table>

- **ITER:** JSP Q, INCHR; INPUT CHARACTER
- **ITER2:** JSP Q, INCHR; INPUT CHARACTER

The instructions above control the iteration process, handling input characters and advancing through iteration levels.
SECTION D. DETERMINES STATEMENT TYPE.

STATEPI: JSP Q, INCHR: INPUT CHARACTER

CAIE A.11
CAIN A.40
SETB G.B
MOVE C.POINT
JRST *2

WRDIN: JSP Q, INCHR: INPUT CHARACTER

CAIE A.13
CAIN A.40
JRST ASSIGN

CAIE A.15
JRST SELECT

CAIN A.40
JRST SELECT

IDPB A.G
CAIG B.4
CAIN A.40
JRST SELECT

JRST SELECT

JRST ASSIGN

JRST LINUM; GO TO LINNUM (SECTION B)

JRST LINUM; GO TO LINNUM (SECTION E)

JRST LINUM; GO TO LINNUM (SECTION B)

SELECT: CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

CAMH G. [ASCI] /TYPE/1

JRST LG; SEE SECTION K
JRST LF; SEE SECTION L
JRST fill: SEE SECTION L
425 003310 254 00 00 0003571 JRST ARRAY  [SEE SECTION F]
426 003311 316 07 00 0003571 JRST ASCII /SCALL/
427 003312 254 00 00 0003929 JRST SCALAR  [SEE SECTION Q]
428 003313 316 07 00 000337 JRST [ASCII /CALL/
429 003314 254 00 00 000310 JRST CALL  [SEE SECTION R]
430 003315 316 07 00 0003481 JRST ASCII /RETURN/
431 003316 254 00 00 000320 JRST RETURN  [SEE SECTION S]
432 003317 316 07 00 000341 JRST ASCII /REM/
433 003318 254 00 00 000322 JRST REMARK  [SEE THIS SECTION]
434 003319 254 00 00 000322 JRST STERR  [IF KEY WORD NOT RECOGNIZED, GO TYPE ERROR MESSAGE]
435 003320 265 17 00 000244 REMARK: JSP Q, INCHR  INPUT CHARACTER
436 003321 302 01 00 000315 JRST END OF LINE
437 003322 254 00 00 000322 JRST REMARK  IND. GO INPUT NEXT CHARACTER
438 003323 265 17 00 000220 JRSP Q, OUTCHR  OUTPUT <><
439 003324 265 17 00 000322 JSP Q, INCHR  INPUT LINE FEED
440 003325 254 20 00 000036 JRST LINNUM  [GO TO LINENUM (SECTION B)]

I. SECTION E. PROCESSES RESERVE STATEMENTS.

003330 602 02 00 000020 RSERV: TRUE FLAGS; 20 JIS THIS A Q-FRAME?
003331 254 20 00 000034 JRST RSERV 2 YES, GO TYPE ERROR MESSAGE
003332 265 17 00 000036 JSP Q, INCHR  INPUT CHARACTER
003333 302 21 00 000011 CAIE A,11
003334 304 21 00 000040 CAIN A,40
003335 254 00 000032 JSRP Q, I I2 [IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER]
003336 265 17 00 000061 JSP Q, II2 INPUT NUMBER OF WORDS RESERVED FOR STUDENTS
003337 254 23 00 000041 JRST +2
003338 265 17 00 000032 JSP Q, INCHR  INPUT CHARACTER
003339 302 01 00 000011 CAIE A,11
003340 304 21 00 000040 CAIN A,40
003341 254 00 000030 JSRP Q, I I J IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
003342 302 21 00 000015 CAIE A,15
003343 254 23 00 000031 JRST RSERV 2 YES, GO TYPE ERROR MESSAGE
003344 265 17 000060 JSP Q, FRMLIN 0 OUTPUT <0>
003345 254 00 000035 LINNUM IGO TO LINENUM (SECTION B)
003346 265 17 00 000126 JSRP Q, FRMLIN [TYPE FRAME AND LINE NOS.]

I. SECTION F. PROCESSES ARRAY STATEMENTS.

I. ARRAY1 TRUE FLAGS; 20 JIS THIS A Q-FRAME?
I. JRST ARRERR 2 YES, GO TYPE ERROR MESSAGE
<table>
<thead>
<tr>
<th>FNC2</th>
<th>FNC3</th>
<th>FNCLST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADJA C,FNC2</td>
<td>ADJA C,0</td>
<td>ASCII /SQR/T</td>
</tr>
<tr>
<td>ADJA C,0</td>
<td>ADJA C,121</td>
<td>ASCII /ABS/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /INT/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SBB/T</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /EXP/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /NLOG/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /LOG/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SIND/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SND/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /TAN/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /COS/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /COSD/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ACOS/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ACOSD/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ASIN/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ASIND/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ATAND/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /STR/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SQR/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /COS/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /TANG/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /NEUT/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SIND/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /SEEN/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /ARCTAN/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /RATAN/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /TAG/</td>
</tr>
<tr>
<td>MOVE C,STACK (G)</td>
<td>MOVE C,TYPE (G)</td>
<td>ASCII /TAG/</td>
</tr>
</tbody>
</table>

FNCLST: ASCII /SQR/T

1. IF CHARACTER IS A SPACE OR TAB, GO_INPUT_NEXT_CHARACTER
2. GO TO POLL
CFRAM  MACRO 47(204)=7 20153  2-AUG-74 PAGE 1-34

1803     027720  265 17 0 00 02713*  JSKP Q,DUTCHRX:OUTPUT 1ST DIGIT OF ADDRESS (BASE 90)
1804     027721  206 22 0 00 02720*  MOVE A,B
1805     027722  265 17 0 00 02720*  JSKP Q,DUTCHRX:OUTPUT 2ND DIGIT OF ADDRESS (BASE 90)
1806     027723  344 36 0 00 02661*  ADAJ F,OUT
1807     027724  251 23 0 00 02720*  SPCKOUT
1808     027725  265 17 0 00 02722*  MOVEI A,20:INPUT SPECIAL VARIABLE CODE <20> IN AC-A
1809     027726  244 23 0 00 02714*  JSKP Q,DUTCHRX:OUTPUT <22>
1810     027727  265 17 0 00 02720*  MOVE A,SPACK (F) :GET ABSOLUTE VALUE OF VARIABLE CODE
1811     027728  334 20 0 00 02661*  JSKP Q,DUTCHRX:OUTPUT CODE FOR THIS VARIABLE
1812     027729  251 23 0 00 02720*  ADAJ F,OUT
1813     027730  201 01 0 00 02232*  ARROUT
1814     027731  265 17 0 00 02727*  JSKP Q,DUTCHRX:OUTPUT CODE
1815     027732  551 01 0 00 02714*  HRR2 A,SPACK (F) :GET ADDRESS OF VARIABLE FROM POLISH STACK
1816     027733  251 23 0 00 02012*  IDIVI A,1090
1817     027734  271 01 0 00 02048*  ADDI A,40
1818     027735  271 02 0 00 02040*  ADDI B,40
1819     027736  265 17 0 00 02732*  JSKP Q,DUTCHRX:OUTPUT 1ST DIGIT OF ADDRESS (BASE 90)
1820     027737  206 23 0 00 02722*  MOVE A,B
1821     027738  265 17 0 00 02737*  JSKP Q,DUTCHRX:OUTPUT 2ND DIGIT OF ADDRESS (BASE 90)
1822     027739  551 01 0 00 02714*  HRR2 A,SPACK (F) :GET NO. OF COLUMNS IN THIS ARRAY
1823     027740  251 23 0 00 00132*  IDIVI A,1090
1824     027741  271 01 0 00 02048*  ADDI A,42
1825     027742  271 02 0 00 02040*  ADDI B,42
1826     027743  265 17 0 00 02732*  JSKP Q,DUTCHRX:OUTPUT 1ST DIGIT OF COLUMN COUNT (BASE 90)
1827     027744  206 23 0 00 02722*  MOVE A,B
1828     027745  265 17 0 00 02737*  JSKP Q,DUTCHRX:OUTPUT 2ND DIGIT OF COLUMN COUNT (BASE 90)
1829     027746  551 01 0 00 02714*  HRR2 A,SPACK (F) :GET NUMBER OF CHARACTERS IN THIS ARRAY
1830     027747  251 23 0 00 02010*  ENDOUT
1831     027748  220 01 0 00 02218*  MOVE A,H:OUTPUT CHARACTER THAT TERMINATED EXPRESSION IN AC-A
1832     027749  254 08 1 00 02163*  JRSJ 0PC5:RETURN

;SECTION P. PLACE VARIABLE IN SYMBOL TABLE AND OUTPUT ITS ASSIGNED ADDRESS;

1833     027750  265 17 0 00 02020*  ASVAR
1834     027751  202 17 0 00 02202*  ASVAR
1835     027752  220 02 0 00 02020*  ASVAR
1836     027753  220 18 0 00 02301*  ASVAR
1837     027754  202 22 0 00 02754*  ASVAR
1838     027755  260 21 0 00 02734*  ASVAR
1839     027756  513 15 0 00 02145*  ASVAR
1840     027757  265 17 0 00 02707*  ASVAR
1841     027758  344 36 0 00 02661*  ASVAR
1842     027759  265 17 0 00 02776*  ASVAR
1843     027760  344 36 0 00 02661*  ASVAR
1844     027761  265 17 0 00 02777*  ASVAR
1845     027762  344 36 0 00 02661*  ASVAR
1846     027763  265 17 0 00 02778*  ASVAR
1847     027764  344 36 0 00 02661*  ASVAR
1848     027765  265 17 0 00 02779*  ASVAR
1849     027766  344 36 0 00 02661*  ASVAR
1850     027767  265 17 0 00 02780*  ASVAR
1851     027768  344 36 0 00 02661*  ASVAR
1852     027769  265 17 0 00 02781*  ASVAR
1853     027770  344 36 0 00 02661*  ASVAR
1854     027771  265 17 0 00 02782*  ASVAR
1855     027772  344 36 0 00 02661*  ASVAR

;SAVE RETURN ADDRESS
;SAVE THE CHARACTER IN AC-A
;SAVE THE INDEX INTO SYMBOL TABLE
;GET SYMBOL FROM SYMBOL TABLE
;GET SYMBOL NAME SAME AS SYMBOL
;YES, JUMP
;GO TO NEXT SYMBOL, IF ANY
;IF NAME NOT FOUND, GO TO NOSYM.
;INPUT <22> IN AC-A
;OUTPUT <22>
;GET ADDRESS OF VARIABLE
;OUTPUT 1ST DIGIT OF ADDRESS (BASE 90)
;OUTPUT 2ND DIGIT OF ADDRESS (BASE 90)
;OUTPUT CHARACTER BACK IN AC-A

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION G. PROCESSES SCALAR STATEMENTS.

SCALAR: JSP Q. INCHR ; INPUT CHARACTER

JSP Q. INCHR ; INPUT CHARACTER
SECTION R. PROCESSES CALL STATEMENTS.

CALL 1 MOVEI A,"CH"

JSP Q,OUTCHR OUTPUT CALL STATEMENT CODE = "CH"

MOVEI G,7

JSP Q,INCHR INPUT CHARACTER

CALL A,11

CAIN A,40

JRT *3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL A,13

CAIN A,15

JRT *2 IF NO LESSON NAME, GO TYPE ERROR MESSAGE

JSP Q,INCHR

JSP Q,OUTCHR OUTPUT NEXT CHARACTER OF LESSON NAME

JRT FNERR1 IF END OF NAME?

JSP Q,OUTCHR OUTPUT CHARACTER

JSP Q,OUTCHR OUTPUT CHARACTER IN AD-A

JSP Q,INCHR INPUT CHARACTER

JSP Q,FRML IN TYPE FRAME AND LINE NOS.

TTCALL 3,CASCII / MORE THAN SIX CHARACTERS IN LESSON NAME?/

JRT LINNUM IGO TO LINNUM (SECTION B)

JSP Q,FRML IN TYPE FRAME AND LINE NOS.

TTCALL 3,CASCII / NO LESSON NAME FOLLOWS CALL /

JRT LINNUM IGO TO LINNUM (SECTION B)

JSP Q,OUTCHR OUTPUT CHARACTER IN AD-A

JSP Q,FRML IN TYPE FRAME AND LINE NOS.
1962 003133 302 0 0 002011
1963 003134 304 0 0 002040
1964 003135 305 0 0 002121
1965 003136 403 0 0 002020
1966 003137 304 0 0 002141
1967 003138 325 17 0 0 002132
1968 003139 301 0 0 002068
1969 003140 323 17 0 0 002076
1970 003141 202 0 0 002147
1971 003142 323 0 0 002029
1972 003143 304 0 0 002159
1973 003144 305 0 0 002141
1974 003145 301 0 0 002082
1975 003146 304 0 0 002174
1976 003147 305 0 0 002144
1977 003148 304 0 0 002059
1978 003149 305 0 0 002160
1979 003150 302 0 0 002079
1980 003151 304 0 0 002160
1981 003152 305 0 0 002054
1982 003153 304 0 0 002154
1983 003154 301 0 0 002054
1984 003155 304 0 0 002131
1985 003156 302 0 0 002135
1986 003157 304 0 0 002172
1987 003158 304 0 0 002154
1988 003159 305 0 0 002154
1989 003160 301 0 0 002154
1990 003161 302 0 0 002020
1991 003162 304 0 0 002160
1992 003163 305 0 0 002054
1993 003164 304 0 0 002154
1994 003165 305 0 0 002154
1995 003166 304 0 0 002154
1996 003167 305 0 0 002054
1997 003168 304 0 0 002154
1998 003169 305 0 0 002054
1999 003170 304 0 0 002154
2000 003171 305 0 0 002054
2001 003172 304 0 0 002154
2002 003173 305 0 0 002054
2003 003174 304 0 0 002154
2004 003175 305 0 0 002054
2005 003176 304 0 0 002154
2006 003177 305 0 0 002054
2007 003178 304 0 0 002154
2008 003179 305 0 0 002054
2009 003180 304 0 0 002154
2010 003181 305 0 0 002054
2011 003182 304 0 0 002154
2012 003183 305 0 0 002054
2013 003184 304 0 0 002154
2014 003185 305 0 0 002054

SECTION 5. PROCESSES_RETURN STATEMENTS.

RETUR!
MOVE B,A
ISAVE THE CHARACTER IN AC-A

J
RETRNA MOVE B,A
ISAVE THE CHARACTER IN AC-A

J
RETRNA MOVE B,A
ISAVE THE CHARACTER IN AC-A

J
RETRNA MOVE B,A
ISAVE THE CHARACTER IN AC-A

J
RETRNA MOVE B,A
ISAVE THE CHARACTER IN AC-A
NO ERRORS DETECTED

PROGRAM BREAK IS 004863

5K CORE USED
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>INEND</td>
<td>574</td>
<td>599</td>
</tr>
<tr>
<td>INFLP</td>
<td>162#</td>
<td>1122</td>
</tr>
<tr>
<td>INPN</td>
<td>1657#</td>
<td>1973</td>
</tr>
<tr>
<td>INPUT</td>
<td>247</td>
<td>566#</td>
</tr>
<tr>
<td>INPUT1</td>
<td>567#</td>
<td>596</td>
</tr>
<tr>
<td>IOD</td>
<td>182#</td>
<td>679</td>
</tr>
<tr>
<td>IODB</td>
<td>184#</td>
<td>1976</td>
</tr>
<tr>
<td>IOT</td>
<td>162#</td>
<td>288</td>
</tr>
<tr>
<td>IP3</td>
<td>585</td>
<td>628#</td>
</tr>
<tr>
<td>IP4</td>
<td>527#</td>
<td>661</td>
</tr>
<tr>
<td>IPER2</td>
<td>527</td>
<td>705#</td>
</tr>
<tr>
<td>IPERR</td>
<td>577</td>
<td>596</td>
</tr>
<tr>
<td>ITI</td>
<td>313#</td>
<td>359</td>
</tr>
<tr>
<td>ITER</td>
<td>239</td>
<td>274</td>
</tr>
<tr>
<td>ITER2</td>
<td>294</td>
<td>325#</td>
</tr>
<tr>
<td>ITLEVE</td>
<td>182#</td>
<td>312</td>
</tr>
<tr>
<td>LEVEL</td>
<td>181#</td>
<td>1613</td>
</tr>
<tr>
<td>LFRAC</td>
<td>1652</td>
<td>1658#</td>
</tr>
<tr>
<td>LINE</td>
<td>1644</td>
<td>238</td>
</tr>
<tr>
<td>LINNO</td>
<td>1944</td>
<td>228</td>
</tr>
<tr>
<td>LNUM</td>
<td>236#</td>
<td>278</td>
</tr>
<tr>
<td>MBC</td>
<td>503</td>
<td>774</td>
</tr>
<tr>
<td>MBDONE</td>
<td>314</td>
<td>354</td>
</tr>
<tr>
<td>LSTM1</td>
<td>322</td>
<td>336</td>
</tr>
<tr>
<td>LSTM2</td>
<td>328</td>
<td>358#</td>
</tr>
<tr>
<td>LSTM3</td>
<td>337</td>
<td>372#</td>
</tr>
<tr>
<td>LPAR</td>
<td>1288</td>
<td>1241#</td>
</tr>
<tr>
<td>MEHERR</td>
<td>531</td>
<td>1424</td>
</tr>
<tr>
<td>NDX</td>
<td>635</td>
<td>674#</td>
</tr>
<tr>
<td>NDAND</td>
<td>1694</td>
<td>1744#</td>
</tr>
<tr>
<td>NOSYM</td>
<td>1844</td>
<td>1856#</td>
</tr>
<tr>
<td>NOTCNT</td>
<td>1752</td>
<td>1785#</td>
</tr>
<tr>
<td>NVR1</td>
<td>226</td>
<td>258</td>
</tr>
<tr>
<td>NVR2</td>
<td>257</td>
<td>292#</td>
</tr>
<tr>
<td>NUMMR</td>
<td>1394</td>
<td>1314#</td>
</tr>
<tr>
<td>NUMOUT</td>
<td>1383</td>
<td>1793#</td>
</tr>
<tr>
<td>DND</td>
<td>859</td>
<td>877</td>
</tr>
<tr>
<td>DNDIM</td>
<td>537</td>
<td>554#</td>
</tr>
<tr>
<td>DPNR</td>
<td>159</td>
<td>1532</td>
</tr>
<tr>
<td>DPNR1</td>
<td>1535</td>
<td>1544#</td>
</tr>
<tr>
<td>DPNR2</td>
<td>1557</td>
<td>1562</td>
</tr>
<tr>
<td>DPNR3</td>
<td>1565</td>
<td>1574</td>
</tr>
<tr>
<td>DPNR4</td>
<td>1584</td>
<td>1592</td>
</tr>
<tr>
<td>DPNR5</td>
<td>1597</td>
<td>1601#</td>
</tr>
<tr>
<td>DFOPT</td>
<td>1624</td>
<td>1642#</td>
</tr>
<tr>
<td>DOSTACK</td>
<td>1630</td>
<td>1626</td>
</tr>
<tr>
<td>DPUI</td>
<td>1772#</td>
<td>1789</td>
</tr>
<tr>
<td>DOUTC</td>
<td>1644</td>
<td>239</td>
</tr>
<tr>
<td>DOUTCH</td>
<td>777</td>
<td>801</td>
</tr>
<tr>
<td>DUOT</td>
<td>979</td>
<td>996</td>
</tr>
<tr>
<td>Variable</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>RLPIA</td>
<td>1595</td>
<td></td>
</tr>
<tr>
<td>RLPI</td>
<td>1599</td>
<td></td>
</tr>
<tr>
<td>RLPLA</td>
<td>1598</td>
<td></td>
</tr>
<tr>
<td>RLPL</td>
<td>1599</td>
<td></td>
</tr>
<tr>
<td>RPAR</td>
<td>1526</td>
<td></td>
</tr>
<tr>
<td>RPTALS</td>
<td>1628</td>
<td></td>
</tr>
<tr>
<td>RSV</td>
<td>442</td>
<td></td>
</tr>
<tr>
<td>RSV2</td>
<td>467</td>
<td></td>
</tr>
<tr>
<td>RSVERR</td>
<td>463</td>
<td></td>
</tr>
<tr>
<td>RSVNUM</td>
<td>460</td>
<td></td>
</tr>
<tr>
<td>RTSRAC</td>
<td>1667</td>
<td></td>
</tr>
<tr>
<td>RV1</td>
<td>1386</td>
<td></td>
</tr>
<tr>
<td>RV2</td>
<td>1389</td>
<td></td>
</tr>
<tr>
<td>RV3</td>
<td>1387</td>
<td></td>
</tr>
<tr>
<td>RVAR</td>
<td>1389</td>
<td></td>
</tr>
<tr>
<td>RVAR1</td>
<td>1371</td>
<td></td>
</tr>
<tr>
<td>SB</td>
<td>415</td>
<td></td>
</tr>
<tr>
<td>SB1</td>
<td>485</td>
<td></td>
</tr>
<tr>
<td>SBERR</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>SBER2</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>SCA2</td>
<td>1911</td>
<td></td>
</tr>
<tr>
<td>SCAERR</td>
<td>1866</td>
<td></td>
</tr>
<tr>
<td>SCALAR</td>
<td>427</td>
<td></td>
</tr>
<tr>
<td>SCALK</td>
<td>1876</td>
<td></td>
</tr>
<tr>
<td>SCAR2</td>
<td>1902</td>
<td></td>
</tr>
<tr>
<td>SELECT</td>
<td>394</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>417</td>
<td></td>
</tr>
<tr>
<td>SGR</td>
<td>1222</td>
<td></td>
</tr>
<tr>
<td>SPC1</td>
<td>1276</td>
<td></td>
</tr>
<tr>
<td>SPC2</td>
<td>1344</td>
<td></td>
</tr>
<tr>
<td>SPCV</td>
<td>1287</td>
<td></td>
</tr>
<tr>
<td>SPCV1</td>
<td>1274</td>
<td></td>
</tr>
<tr>
<td>SPCV2</td>
<td>1318</td>
<td></td>
</tr>
<tr>
<td>SSTACK</td>
<td>1635</td>
<td></td>
</tr>
<tr>
<td>STATY</td>
<td>434</td>
<td></td>
</tr>
<tr>
<td>STERR</td>
<td>1153</td>
<td></td>
</tr>
<tr>
<td>STRAP</td>
<td>1152</td>
<td></td>
</tr>
<tr>
<td>STRING</td>
<td>1264</td>
<td></td>
</tr>
<tr>
<td>STYPE</td>
<td>1857</td>
<td></td>
</tr>
<tr>
<td>SVERR</td>
<td>1234</td>
<td></td>
</tr>
<tr>
<td>SYMDX</td>
<td>524</td>
<td></td>
</tr>
<tr>
<td>SYMTAB</td>
<td>534</td>
<td></td>
</tr>
<tr>
<td>SYMTAB1</td>
<td>534</td>
<td></td>
</tr>
<tr>
<td>TABERR</td>
<td>526</td>
<td></td>
</tr>
<tr>
<td>TP1</td>
<td>720</td>
<td></td>
</tr>
<tr>
<td>TP2</td>
<td>727</td>
<td></td>
</tr>
<tr>
<td>TPER2</td>
<td>817</td>
<td></td>
</tr>
<tr>
<td>TPERR</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>TSLA</td>
<td>747</td>
<td></td>
</tr>
<tr>
<td>TXTTP2</td>
<td>226</td>
<td></td>
</tr>
<tr>
<td>TXTTP1</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>TXTTP2</td>
<td>731</td>
<td></td>
</tr>
<tr>
<td>TXTTP3</td>
<td>228</td>
<td></td>
</tr>
</tbody>
</table>
TITLE AUX4

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT  COMPLETED 6-JUN-74

PART OF LPROS.SHR, VERSION 3

AUX4.MAC CONTAINS A NUMBER OF SUBROUTINES WHICH ARE USED BY OTHER
SUBPROGRAMS IN THE LPROS.SHR HIGH SEGMENT.

CONTENTS OF AUX4.MAC

SECTION A, INPUTS A CHARACTER FROM THE DISK INTO AC-A ON
CHANNEL ONE.

SECTION B, OUTPUTS A CHARACTER FROM AC-A TO THE DISK ON
CHANNEL TWO.

SECTION C, OUTPUTS THE OCTAL NO. IN AC-A TO THE DISK.

SECTION D, OUTPUTS THE INTEGER IN AC-A TO THE DISK.

SECTION E, OUTPUTS THE INTEGER IN AC-A TO A USER'S TERMINAL.

SECTION F, TYPES FRAME AND LINE NOS. IN AN ERROR MESSAGE.

SECTION G, TYPES THE SYMBOL IN AC-G.

SECTION H, INPUTS AN INTEGER FROM THE DISK INTO AC-B.

SECTION I, INPUTS A FLOATING POINT NO. FROM THE DISK INTO AC-B.

SECTION J, OUTPUTS THE FLOATING POINT NO. IN AC-A TO THE DISK.

SECTION K, INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B.

ENTRY POINTS TO AUX4.MAC

FLOUT - FROM CFRAM.MAC (SECTIONS L & O), LOCATED IN SECTION J


FRMLN2 - FROM PROS.MAC (SECTIONS E, F, G, H, I, J, K, M & BB), LOCATED IN SECTION F

II - FROM PROS.MAC (SECTIONS B, C, AA & BB) AND CFRAM.MAC
(SECTIONS H & I), LOCATED IN SECTION H
I12 - FROM PROS.MAC (SECTIONS E, F, H, K & V) AND CFRAM.MAC
(SECTIONS B, E, F, G, H, I, J, K & L), LOCATED IN
SECTION M

I17 - FROM PROS.MAC (SECTION A), LOCATED IN SECTION K

INCHR = FROM PROS.MAC (ALL SECTIONS EXCEPT A, I, O, Q, S, T;
W, Z & CC) AND CFRAM.MAC (ALL SECTIONS EXCEPT A;
M & P), LOCATED IN SECTION A

INFLP = FROM CFRAM.MAC (SECTIONS L & O), LOCATED IN SECTION I

I00 - FROM PROS.MAC (SECTIONS C, V, X & BB) AND CFRAM.MAC
(SECTIONS D, H, I, K & L), LOCATED IN SECTION D

I008 = FROM CFRAM.MAC (SECTION R), LOCATED IN SECTION C

IOT - FROM PROS.MAC (SECTIONS BB & CC) AND CFRAM.MAC (SECTION
B)

DUTCHR = FROM PROS.MAC (ALL SECTIONS EXCEPT A, B, I, O, Q, S;
T, Z & CC) AND CFRAM.MAC (ALL SECTIONS), LOCATED IN
SECTION B

TYPSYM - FROM PROS.MAC (SECTIONS D & P), LOCATED IN SECTION D

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN PRSLOW.MAC)

ERRTOT - TOTAL NO. OF ERRORS DETECTED

EXPNT = BASE TEN EXPONENT

FRNUM = CURRENT FRAME NO.

IBUF = INPUT BUFFER HEADER (3 WORDS)

LINE = CURRENT LINE NO.

OBUF = OUTPUT BUFFER HEADER (3 WORDS)

PC2 = SUBROUTINES STORE RETURN ADDRESS HERE

PC3 = SUBROUTINES STORE RETURN ADDRESS HERE

PDLST = PUSH DOWN LIST FOR StORING ASCII DIGITS OF A

NUMBER BEING OUTPUT.

X

HISEG
ENTRY FLOUT, INFLT, I1T, IODB
ENTRY INCNR, OUTCHR, IOD, I1T, FRMLIN, I1, I12, TYPSYM, FRMLINZ
EXTERN IBUF, OBUF, PC2, PDLST, ERTOT, FRMNUM, LINE, PC3, EXPNT

FLAGS=0
A=1
B=2
C=3
D=4
E=5
F=6
G=7
H=16

KI10=1  IFOR KAI0 CPU, LET KI10=0

POINT: POINT 6, 0

SECTION A. THIS SUBROUTINE INPUTS A CHARACTER FROM THE DISK
ION CHANNEL 1, THE CHARACTER IS PLACED IN AC-A.

INCHR: SQRGE IBUF+2 1DECrement CHARACTER COUNT
JRST PUTBUF: IF BUFFER IS FULL, GO EMPTY IT
JRST AC-A: IF CHARACTER IN AC-A
JST (Q) 1YES, RETURN
GETBUF: IN 1, IF FULL BUFFER
JRST INCHR: IF NO ERROR, GO GET NEXT CHARACTER
JRST (Q) 1RETURN
TDCALL 3,EASCIZ /
DISK INPUT ERROR ON CHANNEL 1 = AUX4

SECTION B. THIS SUBROUTINE OUTPUTS A CHARACTER TO THE DISK
ION CHANNEL 2, THE CHARACTER MUST BE IN AC-A.

OUTCHR: SQRGE OBUF+2 1DECrement CHARACTER COUNT
JRST PUTBUF: IF BUFFER IS FULL, GO EMPTY IT
JRST AC-A: IF CHARACTER IN BUFFER
JST (Q) 1RETURN
PUTBUF: OUT 2, EMBT BUFFER
JRST OUTCHR: IF NO ERROR, GO PUT CHARACTER IN BUFFER
TDCALL 3,EASCIZ /
DISK OUTPUT ERROR ON CHANNEL 2 = AUX4

/1
I. SECTION F. THIS SUBROUTINE TYPES THE FRAME AND LINE NUMBER IN WHICH AN ERROR WAS DETECTED.

213 002064 202 17 0 00 002060  \ FILEN2: MOVEM 0,PC3  \ SAVE RETURN ADDRESS
214 002065 355 00 0 00 002060  \ AOS ERRTOT  \ INCREASE ERROR TOTAL
215 002066 254 02 0 00 002183  \ JRST EL  \戈 TO  \L
216 002067 202 17 0 00 002064  \ FILEN2: MOVEM 0,PC3  \ SAVE RETURN ADDRESS
217 002068 355 00 0 00 002060  \ AOS ERRTOT  \ INCREASE ERROR TOTAL
218 002069 356 21 0 00 002122  \ CAIN A,12  \ END OF LINE?
219 00206A 254 02 0 00 002127  \ JRST EL  \YES, GO TO  \L
220 00206B 332 01 0 00 002304  \ CAIN A,4
221 00206C 356 21 0 00 002336  \ CAIN A,12
222 00206D 254 02 0 00 002121  \ JRST EL  \IF END OF FRAME, GO TO  \L
223 00206E 265 17 0 00 002120  \ JSP 0,INCHR  \ INPUT CHARACTER
224 00206F 302 01 0 00 002121  \ CAIN A,12  \ END OF LINE?
225 002070 254 02 0 00 002176  \ JRST -2  \ NO, GO INPUT NEXT CHARACTER
226 002071 202 17 0 00 002175  \ EL:  \ MOVE A,15
227 002072 265 17 0 00 002013  \ JSP 0,OUTCHR  \ OUTPUT <15>
228 002073 265 17 0 00 002013  \ EL2:  \ TTCALL J,ASCII /

229 202121 251 03 0 00 002462  \ FRAME /2
230 202122 220 31 0 00 002030  \ \ MOVE A,FRNUM  \ GET THE CURRENT FRAME NUMBER
231 202123 265 17 0 00 002051  \ JSP 0,1DT  \ \ TYPE THE FRAME NO.
232 202124 251 03 0 00 002465  \ TTCALL J,ASCII /, LINE /3
233 202125 220 31 0 00 002030  \ \ MOVE A,LINE  \ GET THE CURRENT LINE NUMBER
234 202126 265 17 0 00 002051  \ JSP 0,1DT  \ \ TYPE THE LINE NO.
235 202127 265 17 0 00 002057  \ JRST 0,PC3  \ RETURN

II. SECTION G. THIS SUBROUTINE TYPES ILLEGAL SYMBOLS.

236 002112 220 17 0 00 002363  \ TYPSYN: MOVEM 0,PC2  \ SAVE RETURN ADDRESS
237 002113 251 03 0 00 002467  \ TTCALL J,ASCII / - /1
238 002114 220 33 0 00 002006  \ MOVE A,6
239 002115 220 02 0 00 002022  \ MOVE B,POINT  \ PUT BYTE POINTER IN AC-B
240 002116 134 21 0 00 002002  \ TSI  \ ILOD A,8  \ GET SYMBOL CHARACTER
241 002117 271 01 0 00 003240  \ ADDT A,40  \ CONVERT FROM SIXBIT TO ASCII
242 002118 251 01 0 00 002261  \ TTCALL 1,A  \ \ TYPE THE CHARACTER
243 002119 367 03 0 00 002166  \ SDUG C,TS  \ \ GO GET NEXT CHARACTER, IF ANY
244 00211A 254 02 1 00 002112  \ JRST 0,PC2  \ RETURN

III. SECTION H. THIS SUBROUTINE INPUTS AN INTEGER FROM THE DISK AND PLACES IT IN AC-B.

245 002123 220 17 0 00 002122  \ III: MOVEM 0,PC2  \ SAVE RETURN ADDRESS
246 002124 402 02 0 00 002000  \ SETA B
247 002125 265 17 0 00 002001  \ IIIB: JSP 0,INCHR  \ \ INPUT CHARACTER
248 002126 350 00 0 00 002026  \ AOS F  \ \ INCREASE LESSON CHARACTER COUNT
249 002127 391 01 0 00 002068  \ CAIL A,60
2. Aug. 74, Page 1-6

```assembly
319 222231 252 24 0 0 222222
320 0 022239 0 00 0 002202
321 0 022239 0 00 0 002202
322 0 022239 0 00 0 002202
323 0 022239 0 00 0 002202
324 0 022239 0 00 0 002202
325 0 022239 0 00 0 002202
326 0 022239 0 00 0 002202
327 0 022239 0 00 0 002202
328 0 022239 0 00 0 002202
329 0 022239 0 00 0 002202
330 0 022239 0 00 0 002202
331 0 022239 0 00 0 002202
332 0 022239 0 00 0 002202
333 0 022239 0 00 0 002202
334 0 022239 0 00 0 002202
335 0 022239 0 00 0 002202
336 0 022239 0 00 0 002202
337 0 022239 0 00 0 002202
338 0 022239 0 00 0 002202
339 0 022239 0 00 0 002202
340 0 022239 0 00 0 002202
341 0 022239 0 00 0 002202
342 0 022239 0 00 0 002202
343 0 022239 0 00 0 002202
344 0 022239 0 00 0 002202
345 0 022239 0 00 0 002202
346 0 022239 0 00 0 002202
347 0 022239 0 00 0 002202
348 0 022239 0 00 0 002202
349 0 022239 0 00 0 002202
350 0 022239 0 00 0 002202
351 0 022239 0 00 0 002202
352 0 022239 0 00 0 002202
353 0 022239 0 00 0 002202
354 0 022239 0 00 0 002202
355 0 022239 0 00 0 002202
356 0 022239 0 00 0 002202
357 0 022239 0 00 0 002202
358 0 022239 0 00 0 002202
359 0 022239 0 00 0 002202
360 0 022239 0 00 0 002202
361 0 022239 0 00 0 002202
362 0 022239 0 00 0 002202
363 0 022239 0 00 0 002202
364 0 022239 0 00 0 002202
365 0 022239 0 00 0 002202
366 0 022239 0 00 0 002202
367 0 022239 0 00 0 002202
368 0 022239 0 00 0 002202
369 0 022239 0 00 0 002202
370 0 022239 0 00 0 002202
371 0 022239 0 00 0 002202
```

Copyright © 1974 by the copyright owner. Further reproduction prohibited without permission.
IAUX4 MAC 47 (3041)-3 00156 2-AUG-74 PAGE 1-7

```assembly
376 302256 202 0 0 0 000312
377 302257 164 0 0 0 22152
378 302258 213 20 0 0 000264
379 302260 164 0 0 0 22152
380 302261 259 12 0 0 00021
381 302262 203 02 0 0 080264
382 302262 254 09 1 0 222024
383 302264 231 02 0 0 220722
384 302265 174 04 0 0 22051
385 302266 213 02 0 0 000264
386 302267 174 04 0 0 22051
387 302268 254 06 0 0 080264
388 302271 777777 777766
389 302271 222 17 0 0 000311
390 302272 220 02 0 0 000264
391 302274 322 02 0 0 00421
392 302275 282 02 0 0 000262
393 302276 242 03 0 0 082222
394 302277 282 02 0 0 000264
395 302278 257 35 0 0 202262
396 302279 222 34 0 0 002003
397 302280 231 04 0 0 201750
398 302281 282 34 0 0 002020
399 302282 322 04 0 0 202312
400 302283 321 04 0 0 202312
401 302284 213 02 0 0 000265
402 302285 174 02 0 0 021512
403 302286 174 02 0 0 021512
404 302287 315 02 0 0 202152
405 302288 254 02 0 0 202326
406 302289 174 02 0 0 021512
407 302290 320 02 0 0 000264
408 302291 164 02 0 0 000162
```

The routine below converts the number in AC-B to floating point.

Section 1: This subroutine outputs a floating point number into the disk. The number must be in AC-A.

MINTEN: -12

FLOUTI: MOVEM 0,PC3 ;SAVE RETURN ADDRESS
MOVE B,A ;PUT THE NO. IN AC-B
JUMPE B,TYPER IF NO. IS ZERO, GO TO TYPER
MOVE C,B ;PUT NO. IN AC-C
LSH C,-33 ;SHIFT TO GET ONLY THE EXPONENT IN AC-C
MOVE D,-55 ;PUT LOG10(2)*123 IN AC-D
SUBI C,28 ;SUBTRACT BECAUSE EXPONENT IS EXCESS 128
IMUL D,C ;CONVERT TO EXPONENT OF TEN
IDIV D,01620 ;IGNORE FRACTIONAL PART OF EXPONENT
MOVE D,EXPRT ;SAVE THE EXPONENT
JUMPE D,ZEXP ;JUMP IF EXPONENT IS ZERO
JUMPL D,NEGEXP ;JUMP IF EXPONENT IS NEGATIVE
IDIV D,12 ;SEPARATE EXPONENT INTO TWO PARTS
MOVEM E,IMAKE LOW ORDER PART NEGATIVE
FOVR B,TENTAB-1103 ;DIVIDE TO GET NO. LESS THAN 1.0
ZEREXP: CANGE B,TENTAB+12 && NO. STILL GREATER THAN 1.0?
JRST OUTPUT ;NO. GO TO OUTPUT
FOVR B,TENTAB-11 ELSE, DIVIDE BY 10.0
AOS EXPRT ;ADD ONE TO VALUE OF EXPONENT
JRST OUTPUT ;GO TO OUTPUT
NEGEXP :IDIV D,MINTEN ;SEPARATE EXPONENT INTO TWO PARTS
FMFR B,TENTAB-1(D)
\begin{verbatim}
  478 023373 265 17 0 00 022631] JSP OUTCHR I OUTPUT CHARACTER
  479 023374 322 05 0 00 022421] JUMP E TYPEXP IF NO MORE CHARACTERS, GO TO TYPEXP
  480 023375 366 00 0 00 022935] 2: SP, B, +3 JUMP IF PERIOD DOES NOT GO HERE
  481 023376 211 01 0 00 022935] MOVEI A, "M
  482 023377 269 17 0 00 023013] JSP OUTCHR I OUTPUT DECIMAL POINT
  483 023382 354 05 0 00 022372] SOJA E GETGQT 12 GET NEXT CHARACTER FROM PUSH DOWN LIST
  484 023381 322 03 1 02 022272] TYPEXP: JUMPE C, PC: I RETURN IF EXPONENT IS ZERO
  485 023402 221 01 0 00 022351] MOVEI A, "E"
  486 023403 269 17 0 00 023013] JSP OUTCHR I OUTPUT "E"
  487 023404 327 03 0 00 023410] JUMP C, POSEXP I JUMP IF EXPONENT IS POSITIVE
  488 023405 211 31 0 00 022355] MOVEI A, "M
  489 023406 269 17 0 00 023013] JSP OUTCHR I OUTPUT "M"
  490 023407 217 22 0 00 020035] MOVS C ITAKE ABSOLUTE VALUE OF EXPONENT
  491 023408 231 03 0 00 022712] POEXP: IDIV C, 12 I SEPARATE EXPONENT INTO TWO DIGITS
  492 023409 322 03 0 00 023415] JUMPE C, +4 I JUMP IF 1ST DIGIT IS ZERO
  493 023412 435 03 2 00 022760] IDOR C, 60 I CONVERT TO ASCII CODE
  494 023413 260 21 0 00 022803] MOVE A, C
  495 023414 439 04 0 00 022760] JSP OUTCHR I OUTPUT CHARACTER
  496 023415 439 04 0 00 022760] IDOR C, 60 I CONVERT 2ND DIGIT TO ASCII CODE
  497 023416 203 21 0 00 023024] MOVE A, D
  498 023417 265 17 0 00 023013] JSP OUTCHR I OUTPUT CHARACTER
  499 023418 254 00 1 00 023421] JRST PC: I RETURN
  500 023419 281 01 0 00 023026] MOVEI A, "Q"
  501 023420 269 17 0 00 023013] JSP OUTCHR I OUTPUT A ZERO
  502 023423 254 00 1 00 023420] JRST PC: I RETURN

507

SECTION K. THIS SUBROUTINE INPUTS AN INTEGER FROM A USER'S TERMINAL AND PLACES IT IN AC-B.

509

510 023424 403 02 0 00 023200] IIT: SETE B,
  511 023425 251 24 0 00 028001] TDCALL 4, A I INPUT ASCII CHARACTER
  512 023426 306 21 0 00 022940] CAIN A, +4 I USE CHARACTER A SPACE?
  513 023427 254 02 0 00 023425] JRST +2 I YES, GO INPUT NEXT CHARACTER
  514 023431 254 32 0 00 024321] JRST +2 I RETURN IF CHARACTER IN NOT A DIGIT
  515 023431 251 24 0 00 023021] IIT1: TDCALL 4, A I INPUT NEXT CHARACTER
  516 023432 301 21 0 00 022960] CAIL A, 60 I USE CHARACTER A SPACE?
  517 023433 254 22 0 00 023971] IIT2: CAIL A, 60 I ADD DIGIT TO NO. ALREADY IN AC-B
  518 023434 251 24 0 00 022940] JRST (Q) I ADD DIGIT TO NO. ALREADY IN AC-B
  519 023435 221 22 0 00 020012] IMULI B, 12 I MULTIPLY BY 13
  520 023436 271 02 0 00 0277720] ADDI B, -60(A) I GO GET NEXT DIGIT

523
  523 023437 254 20 0 00 023531] NDSYM
  524 END

NO ERRORS DETECTED

PROGRAM BREAK IS .085472

3K CORE USED
\end{verbatim}
<table>
<thead>
<tr>
<th></th>
<th>ID1</th>
<th>K18</th>
<th>128</th>
<th>232#</th>
<th>205</th>
<th>236</th>
<th>239</th>
<th>335</th>
<th>337</th>
<th>342</th>
<th>344</th>
<th>348</th>
<th>350</th>
<th>387</th>
<th>431</th>
<th>436</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>LINE</td>
<td>112#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MINTEN</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>NEGE</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NOTZER</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>OUTF</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>OUTPUT</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PC2</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>PC3</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>PCST</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>POINT</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>POSOP</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>PUTBUF</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>TENTAB</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>TENTB2</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>TS</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>TYPEXP</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>TYPE2</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>TYPE2</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>REREXP</td>
<td>110#</td>
<td>116</td>
<td>118</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table representing a table of data with columns and rows.*
TITLE PRSLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT COMPLETED 6-JUN-74

LOW SEGMENT FOR LPROS, SHR, VERSION 3

PRSL,MAC IS A DUMMY LOW SEGMENT WHICH IS USED TO LABEL MEMORY LOCATIONS IN THE LOW SEGMENT SO THAT THEY CAN BE EASILY REFERENCED BY PROGRAMS IN THE LPROS HIGH SEGMENT. SINCE NO INSTRUCTIONS OR CONSTANTS ARE STORED IN THE LOW SEGMENT, NO BINARY CORE IMAGE OF IT IS PRODUCED BY THE SSAVE MONITOR COMMAND.

% ENTRY INBUF,OUTBUF,ARGNT,NUMLEV

INTERN PSTACK,PTYPE,HOLD,PC5,EXPNT,RSVNUM,TRMBLK,JOBNO,
INTERN PC4,ADDR,SYMIDX,SYMTPAR,STSTACK,STYPE,OSTACK,ASGN,LEVEL,ITRLEV
INTERN PART,LINE,FRMTOT,ERRTOT,FRNMNUM,FRTYP
INTERN STORE,INAM,ONAM,PC1,IBUF,OBUF,PC2,PDST,PPN,PC3
INTERN DELS,PRST,PREFN,FRMCNT,LINENO,GOTONO

000317 LOC JOBVER

XAD 301,1

RELOC 0

STORE: BLOCK 1 IUSED BY LPROS,MAC
ARGNT: BLOCK 6 IUSED BY LPROS,MAC
JOBNO: BLOCK 7 JOB NUMBER
TRMBLK: BLOCK 8 TRMOP, ARGUMENTS
INTLBLK: BLOCK 9
INTLOC: BLOCK 12
IBUF: BLOCK 3 IUSED BY AUX4,MAC
OBUF: BLOCK 3 IUSED BY AUX4,MAC
ONAM: BLOCK 4 IUSED BY PROS,MAC
PPN: BLOCK 1
INBUF: BLOCK 263
OUTBUF: BLOCK 263

I ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN ILOW,MAC, LDTLOW,MAC, AND RPTLOW,MAC.

PC1: BLOCK 1 IUSED BY PROS,MAC
PC2: BLOCK 1 IUSED BY PROS,MAC AND AUX4,MAC
PC3: BLOCK 1 IUSED BY PROS,MAC, CFRAA,MAC, AND AUX4,MAC
PC4: BLOCK 1 IUSED BY CFRAA,MAC
POS: BLOCK 1 IUSED BY CFRAA,MAC
PDST: BLOCK 10 IUSED BY AUX4,MAC
PART: BLOCK 1 IUSED BY PROS,MAC
LINE: BLOCK 1 IUSED BY PROS,MAC, CFRAM,MAC, AND AUX4,MAC
RESET: BLOCK 1 IUSED BY PROS,MAC
PRSTI: BLOCK 1 IUSED BY PROS,MAC
PRSFN: BLOCK 1 IUSED BY PROS,MAC
ERRTOT: BLOCK 1 IUSED BY PROS,MAC, CFRAM,MAC, AND AUX4,MAC
FRMNUM: BLOCK 1 IUSED BY PROS,MAC, CFRAM,MAC, AND AUX4,MAC
FRGMTYP: BLOCK 1 IUSED BY PROS,MAC
DELS: BLOCK 21 IUSED BY PROS,MAC
LEVEL: BLOCK 4 IUSED BY CFRAM,MAC
ASGN: BLOCK 12 IUSED BY CFRAM,MAC
ADDR: BLOCK 1 IUSED BY PROS,MAC AND CFRAM,MAC
LINENO: BLOCK 1 IUSED BY CFRAM,MAC
GOTOND: BLOCK 1 IUSED BY CFRAM,MAC
SYMIDX: BLOCK 1 IUSED BY PROS,MAC AND CFRAM,MAC
SYMB: BLOCK 140 IUSED BY PROS,MAC AND CFRAM,MAC
SSTACK: BLOCK 112 IUSED BY CFRAM,MAC
STYPE: BLOCK 110 IUSED BY CFRAM,MAC
OSTACK: BLOCK 48 IUSED BY CFRAM,MAC
PSTACK: BLOCK 110 IUSED BY CFRAM,MAC
PTYPE: BLOCK 110 IUSED BY CFRAM,MAC
HOLD: BLOCK 1 IUSED BY CFRAM,MAC
ITRLEV: BLOCK 1 IUSED BY CFRAM,MAC
EXPNT: BLOCK 1 IUSED BY AUX4,MAC
RSVNUM: BLOCK 1 IUSED BY PROS,MAC AND CFRAM,MAC
NUMLST: BLOCK 1 IUSED BY PROS,MAC

NO ERRORS DETECTED

PROGRAM BREAK IS 001412

3K CORE USED
LEARN is a complete system of computer-assisted instruction. Using LEARN, a person with little programming experience can enter CALI lessons into the computer system for later presentation to students located at remote teletypes, terminals can be equipped with slide projector interfaces so that information can be presented to the students on slides.

Lessons must be programmed in LIL, the LEARN Instructional Language. Lessons written in LIL are divided into frames in the same way that a programmed text is divided into frames. There are four types of frames: Q, C, R, and B. Q-frames are used to present questions to students and evaluate their answers. C-frames are used for mathematical calculations. R-frames are used to establish review points in a lesson, and B-frames establish break points in the lesson.

SLIDE 0.10

LEARN also has a calculation mode which can be used by students to solve problems which are presented to them in a lesson. The student enters the calculation mode by typing the ESC or ALT mode key. After completing his calculations, he types ESC (or ALT mode) again to return to the lesson and answer the question.

After a lesson has been typed in by an instructor, it must be processed. During processing, the lesson is checked for programming errors, and it is converted to a form which can be more easily interpreted by the lesson interpreter. A single copy of a lesson can be used by any number of students simultaneously.

Student performance data is stored automatically by the LEARN system. Instructors can generate reports in several different formats to obtain the stored information.

Do you want additional information?

A 0 Yes
B 0 No
PAGE 3

42  ?
43  0
44  ACTIONS
45  A
46  B110
47  B
48  E1
49  Y
50  Rl
51  PLEASE ANSWER YES OR NO.

30 TYPE Q

PAGE

THE FOLLOWING COMMANDS MAY BE TYPED AFTER "ENTER COMMAND".
5

BYE - TO RUN THE LOGOUT PROGRAM
CLEAR - TO INITIALIZE FILES FOR STORING STUDENT
PERFORMANCE DATA
COPY - TO MAKE A COPY OF A LEARN FILE
DELETE - TO DELETE A LEARN FILE
EDIT - TO MAKE CHANGES IN A LESSON
FILES - TO OBTAIN A LIST OF LEARN FILES
WHICH ARE STORED IN YOUR DISK AREA
GET - TO RUN A LEARN LESSON
HELP - TO RUN THIS HELP LESSON
JOB - TO BEGIN THE LOGOUT PROCESS
LIST - TO PRINT A LEARN LESSON ON THE LINE PRINTER
MONITOR - TO RETURN TO MONITOR LEVEL
PROCESS - TO PROCESS A LEARN LESSON
RENAMr - TO CHANGE THE NAME OF A LEARN FILE
REPORT - TO OBTAIN A LEARN REPORT
TEST - TO RUN A LESSON IN TEST MODE
UPDATE - TO UPDATE THE MASTER RECORD FILE
WRITE - TO ENTER A NEW LESSON INTO THE
COMPUTER SYSTEM.

TYPE THE NAME OF THE COMMAND. IF YOU WANT MORE INFORMATION
ON THAT COMMAND, OTHERWISE, TYPE X.

48 TYPE Q

ANSWERS

EXTRA OFF

A 0 BYE

B 0 CLEAR

C 0 COPY

D 0 DELETE

E 0 EDIT

F 0 FILES

G 0 GET

H 0 HELP

I 0 JOB

J 0 LIST

K 0 MONITOR

L 0 PROCESS


PAGE 5

5  ANSWERS
   A 0 YES
   B 0 NO

10 ACTIONS
   A
   B110
   B
   E1
   B

16 RIPLEASE ANSWER YES OR NO.

1 78 TYPE=0
   PAGE
   TEXT
   TYPE THE COMMAND NAME TO GET INFORMATION ON THE COMMAND OF
   YOUR CHOICE. TYPE "COMMANDS" IF YOU WANT TO SEE THE LIST OF
   COMMANDS AGAIN.

8 ACTIONS
   B140

1 88 TYPE=0
   TEXT
   DO YOU WANT INFORMATION ON ANOTHER COMMAND?

3 ANSWERS
   A 0 YES
   B 0 NO

10 ACTIONS
   A
   B170
   B150
   B

19 RIPLEASE ANSWER YES OR NO.

1 98 TYPE=0
   PAGE
   TEXT
   THE BYE COMMAND IS USED TO LOGOUT DIRECTLY WITHOUT GOING
   THROUGH THE STANDARD LOGOUT PROCEQURE.
   THE FORM OF THE BYE COMMAND IS:
   BYE-2
   THERE ARE NO ARGUMENTS FOR THIS COMMAND.

10 ACTIONS
   B150
The first form of the delete command is:

```
DELETE <FILENAME>/<SWITCH> =2
```

Where:

- `<FILENAME>` is the name of the file being deleted.
- `<SWITCH>` is an optional argument that indicates which kind of file is being deleted.
  - `U` = for an unprocessed lesson file
  - `P` = for a processed lesson file
  - `B` = for both unprocessed and processed files
  - `R` = for a record file
  - `A` = for an answer file

If `<SWITCH>` is omitted, an unprocessed file will be deleted.

Examples:

```
DELETE LESSN1/P
DELETE LESSN2/B
DELETE TRANS/R =2
```

The second form of the delete command is:

```
DELETE LESSON=<PPN1>-<PPN2>({<D1>,-<D2>})<CF1>-<CF2>/1 =2
```

Where:

- `<LESSON>` is an optional argument that indicates that information from the specified lesson is to be deleted.
- `<PPN1>` and `<PPN2>` specify that information on students whose programmer nos. are in the range `<PPN1>-<PPN2>` is to be deleted (also optional).
- `<D1>` and `<D2>` specify that information gathered between the dates `<D1>` and `<D2>` is to be deleted (optional).
- `<CF1>` and `<CF2>` specify that information on frames `F1` through `F2` is to be deleted (optional).

The `/*` specifies that information is to be deleted from the master record file `MASTER.REC`:

```
EXAMPLE:
DELETE LESSN1/1
DELETE 2480-2480/1
DELETE LESSN1,2480-2480/1
DELETE LESSN1,2480-2480/12/12/74=3/12/74/1
DELETE LESSN2,1980-1980/1
```

Actions

```
BIBB
```

138 TYPE-Q

The edit command is used to make additions or changes in a lesson which has already been entered into the computer system.

The form of the edit command is:

```
EDIT LESSON=<FILENAMENAME>/<INCREMENT> =2
```

Where:

- `<FILENAME>` is the filename which was assigned to the lesson.
- `<INCREMENT>` is an optional argument which determines the difference between successive frame nos. of any frames which are appended to the lesson.

If `<INCREMENT>` is omitted, the frame increment will be 10.
16  EXAMPLES  EDIT LESSON/S
17  \  
18  ACTIONS  B180  
19  

1  TYPE-O  
2  PAGE  
3  TEXT  
4  THE FILES COMMAND IS USED TO OBTAIN A LISTING OF LEARN  
5  FILES WHICH ARE STORED IN THE USER'S DISK AREA.
6  THE FORM OF THE FILES COMMAND IS:\-
7  FILES/<SWITCH>
8  WHERE:\-
9  <SWITCH> IS AN OPTIONAL ARGUMENT THAT INDICATES THE  
10  KINDS OF FILES TO BE LISTED  
11  U - FOR UNPROCESSED LESSON FILES  
12  P - FOR PROCESSED LESSON FILES  
13  R - FOR RECORD FILES  
14  A - FOR ANSWER FILES
15  IF <SWITCH> IS OMITTED, ALL FILES WILL BE LISTED. MORE THAN  
16  ONE LETTER MAY BE USED FOR THE SWITCH:\-
17  EXAMPLES:  FILES  
18  FILES/U  
19  FILES/P  
20  
21  ACTIONS  B180  
22  

1  TYPE-O  
2  PAGE  
3  TEXT  
4  THE GET COMMAND IS USED BY STUDENTS TO RUN A LESSON WHICH  
5  HAS BEEN PLACED ON THE COMPUTER BY THEIR INSTRUCTOR.
6  THE FORM OF THE GET COMMAND IS:\-
7  GET <LESSON NAME>/<PnP>\-
8  WHERE:\-
9  <LESSON NAME> IS THE FILENAME WHICH HAS BEEN ASSIGNED  
10  TO THE LESSON  
11  <PnP> IS AN OPTIONAL ARGUMENT WHICH CONSISTS OF THE  
12  PROJECT-PROGRAMMER NOS., SEPARATED BY A COMMA UNDER  
13  WHICH THE LESSON IS STORED\-
14  IF <PnP> IS OMITTED, LEARN SEARCHES FOR THE LESSON UNDER  
15  IDENTICAL PROJECT-PROGRAMMER NOS., WHICH ARE EQUAL TO THE USER'S  
16  PROJECT NO. FOR THIS REASON, INSTRUCTORS SHOULD NORMALLY HAVE  
17  IDENTICAL PROJECT-PROGRAMMER NOS., AND STUDENT PROJECT  
18  NOS. SHOULD BE THE SAME AS THEIR INSTRUCTOR'S PROJECT NO.\-
19  EXAMPLES:  GET LESSN  
20  GET LESSN(24020,24025)  
21  
22  ACTIONS  B180  
23  
160 TYPE-Q
PAGE
TEXT
THE HELP COMMAND IS USED TO RUN THIS HELP LESSON.
HELP
THE FORM OF THE HELP COMMAND IS:
HELP
THERE ARE NO ARGUMENTS FOR THIS COMMAND.

Actions
B160

178 TYPE-Q
PAGE
TEXT
THE KJOB COMMAND IS USED TO INITIATE THE LOGOUT PROCESS.
KJOB
KJOB
THE FORM OF THE KJOB COMMAND IS:
KJOB
THERE ARE NO ARGUMENTS FOR THIS COMMAND.

Actions
B160

160 TYPE-Q
PAGE
TEXT
THE LIST COMMAND IS USED TO OBTAIN LESSON LISTINGS ON THE
FILE STORED ON THE
FILE
THE FORM OF THE LIST COMMAND IS:
LIST LESSON
WHERE LESSON IS THE NAME OF THE LESSON
EXAMPLE: LIST LESS2/3

Actions
B160

190 TYPE-Q
PAGE
TEXT
THE MONITOR COMMAND IS USED TO EXIT FROM LEARN AND RETURN TO
THE FORM OF THE MONITOR COMMAND IS:
MONITOR
THERE ARE NO ARGUMENTS FOR THIS COMMAND.

Actions
220 TYPE-Q

PAGE

TEXT

THE PROCESS COMMAND IS USED TO PROCESS LEARN LESSON FILES.

LESSONS MUST BE PROCESSED BEFORE THEY CAN BE USED BY STUDENTS.

THE FORM OF THE PROCESS COMMAND IS:

`PROCESS <LESSON NAME>/<F1>-<F2>`

WHERE:

- `<LESSON NAME>` IS THE FILENAME OF THE LESSON TO BE PROCESSED
- `<F1>` AND `<F2>` ARE OPTIONAL ARGUMENTS THAT INDICATE THAT ALL FRAMES WITH FRAME NOS. BETWEEN F1 AND F2 ARE TO BE PROCESSED.

IF `<F1>` AND `<F2>` ARE OMITTED, THE ENTIRE LESSON WILL BE PROCESSED.

EXAMPLES:

- `PROCESS LSN2`
- `PROCESS LSN2/200`
- `PROCESS LSN4/200-400`

```
\ACTION
```

210 TYPE-Q

PAGE

TEXT

THE RENAME COMMAND IS USED TO CHANGE THE NAMES OF LEARN FILES. THE FILES MUST BE STORED IN THE USER'S OWN DISK AREA.

THE FORM OF THE RENAME COMMAND IS:

`RENAME <NEW NAME>/<OLD NAME>/<SWITCH>`

WHERE:

- `<NEW NAME>` IS THE NEW FILENAME ASSIGNED TO THE FILE
- `<OLD NAME>` IS THE OLD FILENAME
- `<SWITCH>` IS AN OPTIONAL ARGUMENT THAT INDICATES THE KIND OF FILE TO BE RENAMED

- U - FOR AN UNPROCESSED LESSON FILE
- P - FOR A PROCESSED LESSON FILE
- R - FOR A RECORD FILE
- A - FOR AN ANSWER FILE

IF `<SWITCH>` IS OMITTED, AN UNPROCESSED FILE WILL BE RENAMED.

EXAMPLES:

- `RENAME PHYS1=LSN2`
- `RENAME PHA0=MASTER/R`

```
\ACTION
```

220 TYPE-Q

PAGE

TEXT

THE REPORT COMMAND IS USED TO OBTAIN LEARN REPORTS.
THE FORM OF THE REPORT COMMAND IS:

```
REPORT <DEVD> <LESSON>, (<P1>-<P2>) <(D1)-<D2>) <(F1)-<F2>) /S
```

WHERE:

- `<DEVD>` indicates the device on which the report is to be printed and the style of the report. `<DEVD>` may be `TTY` or `LPT` (LINE PRINTER = DETAILED REPORT).
- `<LESSON>` is the filename of the lesson for which a report is desired.
- `<P1>` and `<P2>` are programmer nos. of students.
- `<D1>` and `<D2>` are dates.
- `<F1>` and `<F2>` are frame numbers.
- `<S>` is one of the following switches: `L` for a lesson report, `S` for a student report, or `A` for an answer listing.

All of the arguments are optional.

**EXAMPLES:**

```
REPORT TTYLESN1
REPORT LPTLESN2
REPORT TTYLESN2/A
REPORT TTYLESN3/S
REPORT TTY(2408-2458)/S
REPORT LPTLESN3/(2408-2458)
REPORT TTYLESN5/(2/25/74-3/25/74)
REPORT TTYLESN41
```

**ACTIONS**

```
B158
```

---

230 TYPE-0

**PAGE TEXT**

The test command is used to execute lessons in test mode.

The lesson must be stored in the user's own disk area.

The form of the test command is:

```
TEST <LESSON NAME>/<(F1)> -<F2>)
```

WHERE:

- `<LESSON NAME>` is the filename of the lesson.
- `<F1>` and `<F2>` are frame nos. which indicate the range of frame nos. to be included in the test (optional).

The lesson is processed before execution if the processed lesson file is older than the unprocessed file. The following test mode commands may be typed when Learn is waiting for an answer to any of the questions in the lesson:

```
IN = BRANCH TO FRAME N (E.G., IN 100)
IE = ENTER THE LESSON EDITOR
II = TERMINATE THE LESSON
```

**EXAMPLES:**

```
TEST LESN1
```

**ACTIONS**

```
B158
```

---

240 TYPE-0
THE WRITE COMMAND IS USED TO ENTER A NEW LESSON INTO THE COMPUTER SYSTEM.

THE FORM OF THE WRITE COMMAND IS:

```
WRITE <LESSON NAME> <INCREMENT>
```

WHERE:

- `<LESSON NAME>` is the filename which will be assigned to the lesson.
- `<INCREMENT>` is an optional argument which determines the difference between successive frame nos.

IF `<INCREMENT>` is omitted, the frame increment will be 10.

EXAMPLE: WRITE LESSON 9

```
250 TYPE-Q
```

THE UPDATE COMMAND IS USED TO TRANSFER INFORMATION FROM THE TRANSACTION FILE TO THE MASTER RECORD FILE.

```
UPDATE
```

THERE ARE NO ARGUMENTS FOR THIS COMMAND. INFORMATION MUST BE TRANSFERRED TO THE MASTER FILE IF IT IS TO BE INCLUDED IN A LEARN REPORT.

```
260 TYPE-Q
```

SUMMARY OF LEARN EDIT COMMANDS:

1. **A** = APPEND FRAMES TO END OF LESSON
2. **AN** = APPEND LINE N TO FRAMES N
3. **CN,M** = COPY FRAME N, AND NUMBER IT M
4. **DN** = DELETE FRAME N
5. **DN,N** = DELETE FRAMES N THRU M
6. **DN,P** = DELETE LINE P IN FRAME N
7. **DN,P-Q** = DELETE LINES P THRU Q IN FRAME N
8. **H** = TYPE THIS SUMMARY
9. **IN** = INSERT FRAME N
10. **IN,P** = INSERT IN FRONT OF LINE P IN FRAME N
11. **KN** = ADD N TO ALL FRAME NOS, STARTING AT FRAME N
12. **KN,M** = ADD N TO ALL FRAME NOS, STARTING AT FRAME M
13. **MN,N** = RENUMBER FRAME M TO N
14. **RN** = REPLACE FRAME N
15. **RN,P** = REPLACE LINE P IN FRAME N
16. **RN,P-Q** = REPLACE LINES P THRU Q IN FRAME N
PAGE 13

1. TYPE-Q

2. PAGE 6

3. TEXT

4. THE CALCULATION MODE ALLOWS STUDENTS AND LESSON AUTHORS TO USE

5. THE COMPUTER AS A POWERFUL CALCULATOR. THE CALCULATION MODE MAY

6. BE ENTERED BY TYPING THE "ESC" (OR ALT MODE) KEY. THIS MAY

7. BE DONE AFTER LEARN TYPES "ENTER COMMAND" OR DURING A LESSON.

8. FOLLOWING THE ASTERISK WHICH INDICATES THAT LEARN IS READY

9. TO ACCEPT AN ANSWER, AFTER CALCULATIONS ARE COMPLETE, THE USER

10. AGAIN TYPES THE "ESC" (OR ALT MODE) KEY TO GET OUT OF THE CALC-

11. ULA TION MODE. +2

12. THE FOLLOWING MATHEMATICAL OPERATIONS MAY BE PERFORMED: +2

13. ADDITION 3

14. SUBTRACTION =

15. MULTIPLICATION .

16. DIVISION /

17. EXPONENTIATION **

18. THE FOLLOWING FUNCTIONS ARE ALSO AVAILABLE: +2

19. ABS 2

20. INT 1

21. SQRT 0

22. LOG 4

23. EXP 5

24. SIN 6

25. COS 8

26. COSD 9

27. TAN 10

28. ATAND 11

29. ASIN 12

30. ACOS 13

31. ATAN 14

32. ASIND 15

33. ACOSD 16

34. ATAN D

35. THE MOST COMMONLY USED CALCULATION MODE STATEMENT IS THE "TYPE"

36. 72
STATEMENT. THE "TYPE" STATEMENT INSTRUCTS THE MACHINE TO TYPE THE RESULTS OF CALCULATIONS. HERE ARE SOME EXAMPLES:

\[ 2.5e25 \quad \text{(2.5 times 10 to the 25th power)} \]
\[ 1.7e-5 \quad \text{(1.7 times 10 to the minus 5th power)} \]

EXAMPLES BELOW:

\[ \text{NUMBER B} \quad \text{EXAMPLES} \]
\[ 2.5e25 \quad \text{(2.5 times 10 to the 25th power)} \]
\[ 1.7e-5 \quad \text{(1.7 times 10 to the minus 5th power)} \]

SEE THE "LEARN LESSON AUTHOR'S GUIDE" FOR MORE ADVANCED CALCULATION MODE FEATURES.

WOULD YOU LIKE TO WORK SOME EXAMPLE PROBLEMS?

\[ \text{PLEASE ANSWER YES OR NO.} \]

\[ 230 \quad \text{TYPE-O} \]
\[ \text{PAGE} \]
\[ \text{TEXT} \]

HOW MUCH IS 3479 TIMES 345?

REMEMBER TO TYPE THE "ESC" (OR ALT Mode) KEY TO GET INTO THE CALCULATION MODE, THEN TYPE 3479*345 TO COMPUTE THE ANSWER.

FINALLY, TYPE THE "ESC" (OR ALT Mode) KEY TO GET OUT OF CALCULATION MODE. NOW GO AHEAD AND DO THE PROBLEM.

\[ \text{PLEASE ANSWER YES OR NO.} \]

\[ 290 \quad \text{TYPE-O} \]
\[ \text{PAGE} \]
\[ \text{TEXT} \]

WHAT IS THE SQUARE ROOT (SQR) OF 2347?
ANSWER
1 + SQRT(256)
ACTION
1
FI
?
RT
*
FI
AI

302 TYPE-Q
PAGE
TEXT
HOW MUCH IS 5.6E24 DIVIDED BY THE QUANTITY 2000 TIMES 37.297
\$
 ANSWER
0 S*0.01*SA
1 * 5.6E24/(2000*37.29)
2 = 5.6E24/(2000*37.29
ACTION
FI
B150
2
FI
THE PROPER CALC MODE STATEMENT IS
TYPE 5.6E24/(2000*37.29)
YOU FORGOT TO USE PARENTHESES. TRY AGAIN.

310 TYPE-Q
PAGE
TEXT
LEARN LESSONS MUST BE WRITTEN IN LIL - THE LEARN INSTRUCTIONAL
LANGUAGE. LESSONS WRITTEN IN LIL ARE COMPOSED OF UNITS CALLED
FRAMES. THE FOUR TYPES OF FRAMES ARE Q-FRAMES (QUESTION),
C-FRAMES (CALCULATION), R-FRAMES (REVIEW), AND B-FRAMES
(BREAK). A COMPLETE DESCRIPTION OF LIL CAN BE FOUND IN THE
LEARN LESSON AUTHORS GUIDE.
ACTION
B150