8-1974

The Learn System for Computer-Assisted Instruction

Garret Alan Vander Lugt
Western Michigan University

Follow this and additional works at: http://scholarworks.wmich.edu/dissertations

Part of the Education Commons

Recommended Citation
http://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
LEARN LESSON AUTHOR'S GUIDE

by

Garret Alan Vander Lugt

Western Michigan University
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
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 KI10 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.

Kalamazoo
August, 1974

G. V. L.
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 thru 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.

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

| ACKNOWLEDGEMENTS | .................................................. | i |
| PREFACE | .................................................. | ii |

## CHAPTER 1

1.1 Timesharing ........................................ 1-1
1.2 Peripheral Devices .................................. 1-2
1.3 Access to Learn .................................... 1-2
1.4 Programming Capabilities ......................... 1-3
1.5 The Calculation Mode ................................ 1-3
1.6 Record Keeping Capabilities ..................... 1-3
1.7 Test Mode Features .................................. 1-4
1.8 Student Use of Learn ............................... 1-4
1.9 Some Technical Information ....................... 1-4

## CHAPTER 2

2.1 Entering Lessons into the Computer System ........ 2-1
2.2 Processing a Lesson ............................... 2-2
2.3 Setting Up Record Files ............................ 2-3
2.4 Using Learn Lessons ............................... 2-4
2.5 Generating Learn Reports ........................... 2-4

## CHAPTER 3

3.1 The HELP Command .................................. 3-1
3.2 Lesson Editing Commands ........................... 3-2
3.2.1 The WRITE Command .............................. 3-2
3.2.2 The EDIT Command ................................ 3-3
3.3 The PROCESS Command .............................. 3-5
3.4 Lesson Execution Commands ...................... 3-7
3.4.1 The GET Command ................................ 3-7
3.4.2 The TEST Command ................................ 3-8
3.5 File Maintenance Commands ...................... 3-9
3.5.1 The COPY Command ................................ 3-10
3.5.2 The DELETE Command ............................ 3-12
3.5.3 The FILES Command .............................. 3-16
3.5.4 The LIST Command ................................ 3-17
3.5.5 The RENAME Command ............................ 3-18
3.6 Record Keeping Commands ......................... 3-20
3.6.1 The CLEAR Command ................................ 3-20
3.6.2 The UPDATE Command ............................ 3-22
3.6.3 The REPORT Command ............................ 3-23
**TABLE OF CONTENTS**

(continued)

<table>
<thead>
<tr>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-26</td>
</tr>
<tr>
<td>3-26</td>
</tr>
<tr>
<td>3-27</td>
</tr>
</tbody>
</table>

**CHAPTER 3**

LEARN SYSTEM COMMANDS (continued)

<table>
<thead>
<tr>
<th>3.7.1</th>
<th>The MONITOR Command</th>
<th>3-26</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7.2</td>
<td>The KJOB Command</td>
<td>3-26</td>
</tr>
<tr>
<td>3.7.3</td>
<td>The BYE Command</td>
<td>3-27</td>
</tr>
</tbody>
</table>

**CHAPTER 4**

EDITING LEARN LESSONS

<table>
<thead>
<tr>
<th>4.1</th>
<th>Entering the Learn Lesson Editor</th>
<th>4-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Line Numbers</td>
<td>4-1</td>
</tr>
<tr>
<td>4.3</td>
<td>Learn Edit Commands</td>
<td>4-2</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The A Command</td>
<td>4-2</td>
</tr>
<tr>
<td>4.3.2</td>
<td>The C Command</td>
<td>4-3</td>
</tr>
<tr>
<td>4.3.3</td>
<td>The D Command</td>
<td>4-3</td>
</tr>
<tr>
<td>4.3.4</td>
<td>The I Command</td>
<td>4-3</td>
</tr>
<tr>
<td>4.3.5</td>
<td>The K Command</td>
<td>4-4</td>
</tr>
<tr>
<td>4.3.6</td>
<td>The N Command</td>
<td>4-5</td>
</tr>
<tr>
<td>4.3.7</td>
<td>The R Command</td>
<td>4-5</td>
</tr>
<tr>
<td>4.3.8</td>
<td>The T Command</td>
<td>4-6</td>
</tr>
<tr>
<td>4.3.9</td>
<td>The X Command</td>
<td>4-7</td>
</tr>
<tr>
<td>4.4</td>
<td>Exiting from the Learn Lesson Editor</td>
<td>4-7</td>
</tr>
<tr>
<td>4.5</td>
<td>Editing Efficiency</td>
<td>4-8</td>
</tr>
<tr>
<td>4.6</td>
<td>An Example of Lesson Editing</td>
<td>4-8</td>
</tr>
<tr>
<td>4.7</td>
<td>Summary of Learn Edit Commands</td>
<td>4-11</td>
</tr>
</tbody>
</table>

**CHAPTER 5**

THE _LEARN_ INSTRUCTIONAL LANGUAGE

<table>
<thead>
<tr>
<th>5.1</th>
<th>LIL Frames</th>
<th>5-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.2</td>
<td>Q-frames</td>
<td>5-1</td>
</tr>
<tr>
<td>5.2.1</td>
<td>The TEXT Section</td>
<td>5-2</td>
</tr>
<tr>
<td>5.2.1.1</td>
<td>The TEXT Statement</td>
<td>5-2</td>
</tr>
<tr>
<td>5.2.1.2</td>
<td>The SLIDE Statement</td>
<td>5-3</td>
</tr>
<tr>
<td>5.2.1.3</td>
<td>The PAGE Statement</td>
<td>5-4</td>
</tr>
<tr>
<td>5.2.1.4</td>
<td>TEXT Section Switches</td>
<td>5-4</td>
</tr>
<tr>
<td>5.2.1.4.1</td>
<td>The CALC Statement</td>
<td>5-5</td>
</tr>
<tr>
<td>5.2.1.4.2</td>
<td>The HINT Statement</td>
<td>5-5</td>
</tr>
<tr>
<td>5.2.1.4.3</td>
<td>The REVIEW Statement</td>
<td>5-5</td>
</tr>
<tr>
<td>5.2.1.4.4</td>
<td>The RECORD Statement</td>
<td>5-6</td>
</tr>
<tr>
<td>5.2.1.4.5</td>
<td>The STORE Statement</td>
<td>5-6</td>
</tr>
<tr>
<td>5.2.1.4.6</td>
<td>The BOTH Statement</td>
<td>5-6</td>
</tr>
<tr>
<td>5.2.2</td>
<td>The ANSWER Section</td>
<td>5-6</td>
</tr>
<tr>
<td>5.2.2.1</td>
<td>The ANSWER Statement</td>
<td>5-6</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 5</th>
<th>THE <em>LEARN</em> INSTRUCTIONAL LANGUAGE (continued)</th>
</tr>
</thead>
<tbody>
<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>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>CHAPTER 5</th>
<th>THE LEARN INSTRUCTIONAL LANGUAGE (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3.2.7</td>
<td>Mathematical Functions</td>
</tr>
<tr>
<td>5.3.2.8</td>
<td>Special Functions</td>
</tr>
<tr>
<td>5.3.2.8.1</td>
<td>The SEEN Function</td>
</tr>
<tr>
<td>5.3.2.8.2</td>
<td>The RIGHT, WRONG, and NEUT Functions</td>
</tr>
<tr>
<td>5.3.2.8.3</td>
<td>The ATIME Function</td>
</tr>
<tr>
<td>5.3.2.8.4</td>
<td>The PATH Function</td>
</tr>
<tr>
<td>5.3.2.8.5</td>
<td>The TAG Function</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Variable Definition Statements</td>
</tr>
<tr>
<td>5.3.3.1</td>
<td>The SCALAR Statement</td>
</tr>
<tr>
<td>5.3.3.2</td>
<td>The ARRAY Statement</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Assignment Statements</td>
</tr>
<tr>
<td>5.3.5</td>
<td>Input Statements</td>
</tr>
<tr>
<td>5.3.5.1</td>
<td>The INPUT Statement</td>
</tr>
<tr>
<td>5.3.5.2</td>
<td>The FILL Statement</td>
</tr>
<tr>
<td>5.3.6</td>
<td>Output Statements</td>
</tr>
<tr>
<td>5.3.6.1</td>
<td>The TYPE Statement</td>
</tr>
<tr>
<td>5.3.6.2</td>
<td>The ATYPE Statement</td>
</tr>
<tr>
<td>5.3.7</td>
<td>Branching Statements</td>
</tr>
<tr>
<td>5.3.7.1</td>
<td>The GOTO Statement</td>
</tr>
<tr>
<td>5.3.8</td>
<td>The IF Statement</td>
</tr>
<tr>
<td>5.3.9</td>
<td>Calling External Lessons</td>
</tr>
<tr>
<td>5.3.9.1</td>
<td>The CALL Statement</td>
</tr>
<tr>
<td>5.3.9.2</td>
<td>The RETURN Statement</td>
</tr>
<tr>
<td>5.3.10</td>
<td>The RESERVE Statement</td>
</tr>
<tr>
<td>5.3.11</td>
<td>The REM Statement</td>
</tr>
<tr>
<td>5.3.12</td>
<td>Example C-frames</td>
</tr>
<tr>
<td>5.4</td>
<td>R-frames</td>
</tr>
<tr>
<td>5.5</td>
<td>B-frames</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>CHAPTER 7</th>
<th>TESTING LEARN LESSONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Entering Test Mode .......... 7-1</td>
</tr>
<tr>
<td>7.2</td>
<td>Test Mode Commands .......... 7-1</td>
</tr>
<tr>
<td>7.2.1</td>
<td>The :n Command .......... 7-1</td>
</tr>
<tr>
<td>7.2.2</td>
<td>The :E Command .......... 7-2</td>
</tr>
<tr>
<td>7.2.3</td>
<td>The :: Command .......... 7-2</td>
</tr>
<tr>
<td>7.3</td>
<td>An Example .......... 7-2</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 .......... 8-1</td>
</tr>
<tr>
<td>8.2</td>
<td>Calculation Mode Statements .......... 8-1</td>
</tr>
<tr>
<td>8.3</td>
<td>Some Restrictions .......... 8-2</td>
</tr>
<tr>
<td>8.4</td>
<td>Some Example Calculations .......... 8-2</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
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 \@, 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.

1-1
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

*  

1-2
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 8)
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 >A + 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 >A + 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.
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 LESN↓
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 ↓ ...the student runs Learn

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

ENTER COMMAND
*GET LESN1 ↓ ...the student GETs 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 ↓ ...student types BYE to log-off

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.
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
*UPDATE

MASTER FILE UPDATED
```

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.
3.2 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.

3-4
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.
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

3-6
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 [575,600] \...

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

**********

ENTER COMMAND
*GET BIOL [575,600] \...

THE THEORY OF...

**********

ENTER COMMAND
*GET ATOM [360,425] \...

**********

3.4.2 The TEST Command

Function

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

Format

TEST lesson \{ /n \}

3-8
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

\[
\text{COPY file2=file1 } \left\{ \begin{array}{c}
\text{/U} \\
\text{/P} \\
\text{/R} \\
\text{/A}
\end{array} \right. \\
\]

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

3-10

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
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/y1) \}
   \{ /P-q \} \{ (ml/dl/y1-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/y1 - 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.

3-14
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
3.5.3 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

\[
\text{FILES} \left\{ \begin{array}{l}
/U \\
P \\
/R \\
/A \\
\end{array} \right. 
\]

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.
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
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 /

LIST FILE IN PRINT QUEUE / COPIES: 1

ENTER COMMAND *LIST LESN2/3 

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

\[
\text{RENAME name2=namel } \begin{cases} /U \\ /P \\ /R \\ /A \end{cases}
\]

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
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
*RENAMES=ROCKS=LESN1

UNPROCESSED LESSON RENAMED

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

ENTER COMMAND
*RENAMES=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 (<447>) 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

... user clears the answer file for LESN3

ENTER COMMAND

*CLEAR LESN3 J

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

REPORT \{ TTY: \} \{ TTL: \} \{ LPT: \} \{ lesson \} \{ p\} \{ (m1/d1/y1) \} \{ (m1/d1/y1-m2/d2/y2) \} \{ [f1] \} \{ /A \} \{ /S \} \{ /L \}

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)

m1/d1/y1 - 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)

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 \texttt{C} Command

The \texttt{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 \texttt{C} command is:

\texttt{Cn,m}

where \texttt{n} is the number of the frame being copied and \texttt{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.

\texttt{C30,650} \hspace{1cm} \ldots \text{user types copy command}

\texttt{>} \hspace{1cm} \ldots \text{Learn is ready to accept the next command}

4.3.3 The \texttt{D} Command

The \texttt{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 \texttt{DELETE} command for this). The four forms of the \texttt{D} command are:

1) \texttt{Dn}
2) \texttt{Dn-m}
3) \texttt{Dn,p}
4) \texttt{Dn,p-q}

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

\texttt{D190,13-15} \hspace{1cm} \ldots \text{user types command to delete lines 13-15 of frame 190}

\texttt{>} \hspace{1cm} \ldots \text{Learn is ready to accept the next command}

4.3.4 The \texttt{I} Command

The \texttt{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 \texttt{I} command are:

1) \texttt{In}  
2) \texttt{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.

\begin{verbatim}
>160,5 \texttt{I} \hfill \ldots\text{user types insert command}
\hline
5 >B - WASHINGTON \hfill \ldots\text{user inserts lines}
\hline
6 >C - OREGON \hfill \ldots\text{user types ESC to terminate insertion}
\hline
7 >$ \hfill \ldots\text{Learn is ready to accept the next command}
\end{verbatim}

4.3.5 The \texttt{K} Command

The \texttt{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 \texttt{K} command are:

1) \texttt{Kn}  
2) \texttt{Kn,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.

\begin{verbatim}
>10,100 \texttt{K} \hfill \ldots\text{user types \texttt{K} command}
\end{verbatim}

4-4
4.3.6 The \texttt{N} Command

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

\texttt{Nm,n}

where \texttt{m} is the old frame number and \texttt{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.

\texttt{>N30,45} \ldots user types \texttt{N} command

\textnormal{>} \ldots Learn is ready to accept the next command

4.3.7 The \texttt{R} Command

The \texttt{R} command is used to replace selected frames, lines, or character strings in a lesson. The four forms of the \texttt{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 \texttt{n}, the second form is used to replace line \texttt{p} in frame \texttt{n}, and the third form is used to replace lines \texttt{p} through \texttt{q} in frame \texttt{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 \texttt{R} command is shown below.

\texttt{>R250,12} \ldots user types replace command

\texttt{12 > ANSWER 30} \ldots user types new line 12

\textnormal{>} \ldots Learn is ready to accept the next command

The fourth form of the \texttt{R} command is used to replace the old character string in line \texttt{p} of frame \texttt{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 \( \text{!IE.} \) characters as shown in the example below.

\[ \text{>R800,15,,AND } \downarrow \]...

...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.

\[ \text{>R750,5,THR,THE } \downarrow \]...

...user types command to replace THR with THE

\[ > \]...

...Learn is ready to accept the next command

4.3.8 The \text{T} Command

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

1) \text{T}
2) \text{Tn}
3) \text{Tn-m}
4) \text{Tn,p}
5) \text{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 \text{T} command is shown below.

\[ \text{>T300,13-15 } \downarrow \]...

...user types \text{T} command

\begin{verbatim}
13 ANSWER
14 A + NEWTON
15 B - GALILEO
\end{verbatim}...

...Learn types lines

\[ > \]...

...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.

\[\text{\_\text{\textless}x5\text{\_\textgreater}} \]

...user types X command

\[>\text{\_} \]

...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.

\[\text{\_\textless}E\text{\_\textgreater} \]

...user types E command

\[\text{\_\textless}\text{ENTER COMMAND} \]

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

Example 2.

\[\text{\_\textless}EP\text{\_\textgreater} \]

...user types EP command

\[\text{\_\textless}\text{TOTAL ERRORS DETECTED: 0} \]

...no errors were detected during lesson processing

4-7
Example 3.

\[ >\text{ET} \]

...user types \( \text{ET} \) command

TOTAL ERRORS DETECTED: 0

START AT FRAME

...Learn waits for user to type frame number

Example 4.

\[ >\text{M} \]

...user types \( \text{M} \) command

EXIT

...user is at monitor level

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 mini-
mized 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

\[ \text{ENTER COMMAND} \]

\[ * \]

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

\[ >\text{ET} \]

...user types \( \text{ET} \) command

\[ \text{TOTAL ERRORS DETECTED: 0} \]

\[ \text{START AT FRAME} \]

...Learn waits for user to type frame number

Example 4.

\[ >\text{M} \]

...user types \( \text{M} \) command

EXIT

...user is at monitor level

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 mini-
mized 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

\[ \text{ENTER COMMAND} \]

\[ * \]

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

\[ >\text{ET} \]

...user types \( \text{ET} \) command

\[ \text{TOTAL ERRORS DETECTED: 0} \]

\[ \text{START AT FRAME} \]

...Learn waits for user to type frame number

Example 4.

\[ >\text{M} \]

...user types \( \text{M} \) command

EXIT

...user is at monitor level

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 mini-
mized 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

\[ \text{ENTER COMMAND} \]

\[ * \]

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

\[ >\text{ET} \]

...user types \( \text{ET} \) command

\[ \text{TOTAL ERRORS DETECTED: 0} \]

\[ \text{START AT FRAME} \]

...Learn waits for user to type frame number

Example 4.

\[ >\text{M} \]

...user types \( \text{M} \) command

EXIT

...user is at monitor level

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 mini-
mized 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

\[ \text{ENTER COMMAND} \]

\[ * \]

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

\[ >\text{ET} \]

...user types \( \text{ET} \) command

\[ \text{TOTAL ERRORS DETECTED: 0} \]

\[ \text{START AT FRAME} \]

...Learn waits for user to type frame number

Example 4.

\[ >\text{M} \]

...user types \( \text{M} \) command

EXIT

...user is at monitor level

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 mini-
mized 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.
702 TYPE-> Q ↓
1 > TEXT
2 > ENERGY IS THE ABILITY TO DO
3 > ANSWER
4 > PHONIC ON
5 > A + WORK
6 > ACTION
7 > A
8 > F:
9 > ?
10 > Q:
11 > @
12 > F:
13 > A:
14 > G:
15 > F:
16 > A:
17 > $ ...

...user types ESC to end the frame

704 TYPE-> Q ↓
1 > TEXT
2 > WORK IS EQUAL TO FORCE TIMES
3 > ANSWER
4 > A - DISTANCE
5 > B - MASS
6 > ACTION
7 > A
8 > F:
9 > B
10 > F:
11 > ?
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

...user types ESC to end the frame

706 TYPE-> Q ↓
1 ...
2 ...
3 ...
4 ...
5 ...
6 ...
7 ...
8 ...
9 ...
10 ...
11 ...
12 ...
13 ...
14 ...
15 ...
16 ...
17 ...

...user continues to add frames to the lesson

712 TYPE-> $ ...
1 >ET ...

...user types ESC key to stop appending frames

...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.  

> R704,6,-,+  

>ET  

TOTAL ERRORS DETECTED: 0

START AT FRAME*704

FM.704

WORK IS EQUAL TO FORCE TIMES*DISTANCE.

RIGHT!

FM.706

WORK IS A*:: QUANTITY. 

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

4-11
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:

\[ n \text{ TYPE}-t \]

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

\[ n \text{ TYPE}-Q \]
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

(a) \( \text{TEXT text}\ \backslash \)

(b) \( \text{TEXT$ text}\ \backslash \)

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.

\[
\text{TEXT}
\text{WHAT IS THE CAPITAL OF MICHIGAN}\backslash
\]

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.

(a) \( \leftarrow \)

(b) \( \leftarrow n \)

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 re-
turn 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 state-
ment. (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] \?
```

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 charac-
ters 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 be-
ing 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.

5-3
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
SLIDE 5
SLIDE ON
TEXT
ACCORDING TO THIS TABLE, WHAT IS THE HARDNESS OF TALC?

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>

5-4
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

5-5
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
```
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

```
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.
5.2.3.5 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
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

\[ P:\text{...} \]

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:ON
(b) S: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 MASTERED 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.
?
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 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 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.
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

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

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.

5-20
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) \([\text{index}=\text{limit1,limit2}]\)

(b) \([\text{index1}=\text{limit1,limit2} \text{ index2}=\text{limit3,limit4}]\)

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.

5-22
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
@1
@?
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
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.

\[ + \quad \text{for addition} \]
\[ - \quad \text{for subtraction} \]
\[ * \quad \text{for multiplication} \]
\[ / \quad \text{for division} \]
\[ ^{\uparrow} \quad \text{for exponentiation} \]

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.

\[
\begin{align*}
2\times 3 + 4 & \quad \text{(equals 10)} \\
2 \times (3+4) & \quad \text{(equals 14)} \\
6/3 + 3 & \quad \text{(equals 5)} \\
6/(3+3) & \quad \text{(equals 1)} \\
2 \times 2 \times 2 & \quad \text{(equals 8)} \\
2 \times (2+2) & \quad \text{(equals 16)} \\
(3\times (1+2))/(2+4) & \quad \text{(equals 1.5)}
\end{align*}
\]

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.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS</td>
<td>absolute value</td>
</tr>
<tr>
<td>INT</td>
<td>integer</td>
</tr>
<tr>
<td>SQRT</td>
<td>square root</td>
</tr>
<tr>
<td>CBRT</td>
<td>cube root</td>
</tr>
<tr>
<td>EXP</td>
<td>exponential (e^x)</td>
</tr>
<tr>
<td>NLOG</td>
<td>natural logarithm</td>
</tr>
<tr>
<td>LOG</td>
<td>common logarithm</td>
</tr>
<tr>
<td>COS</td>
<td>cosine (argument in radians)</td>
</tr>
<tr>
<td>COSD</td>
<td>cosine (argument in degrees)</td>
</tr>
<tr>
<td>SIN</td>
<td>sine (argument in radians)</td>
</tr>
<tr>
<td>SIND</td>
<td>sine (argument in degrees)</td>
</tr>
<tr>
<td>TAN</td>
<td>tangent (argument in radians)</td>
</tr>
<tr>
<td>TAND</td>
<td>tangent (argument in degrees)</td>
</tr>
<tr>
<td>ACOS</td>
<td>inverse cosine (radians)</td>
</tr>
<tr>
<td>ACOSD</td>
<td>inverse cosine (degrees)</td>
</tr>
<tr>
<td>ASIN</td>
<td>inverse sine (radians)</td>
</tr>
<tr>
<td>ASIND</td>
<td>inverse sine (degrees)</td>
</tr>
<tr>
<td>ATAN</td>
<td>inverse tangent (radians)</td>
</tr>
<tr>
<td>ATAND</td>
<td>inverse tangent (degrees)</td>
</tr>
</tbody>
</table>

Example expressions that contain functions are shown below.

\[
\begin{align*}
\text{SQRT}(900)+X & \\
\text{EXP}(0.67\times \text{NLOG}(X+Y)) &
\end{align*}
\]
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) SEEN(f)
(b) SEEN(f&g)
(c) 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 nth execution of frame f. For example, if frame 100 was executed twice, then

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.

RIGHT(f)  WRONG(f)  NEUT(f)
RIGHT(f&g)  WRONG(f&g)  NEUT(f&g)
RIGHT(f&g&n)  WRONG(f&g&n)  NEUT(f&g&n)

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.

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

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

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.

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

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.

5-29
\[
A = X + 2Y \\
X = \text{SQRT}(A)/N \\
AX(I) = BX(I) \times CX(I)
\]

5.3.5 Input Statements

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

5.3.5.1 The \texttt{INPUT} Statement

The form of the \texttt{INPUT} statement is

\begin{verbatim}
INPUT list
\end{verbatim}

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

\begin{verbatim}
INPUT A, X, AA(I)
\end{verbatim}

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.

\begin{verbatim}
[I=1,10] INPUT AX(I)
\end{verbatim}

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.

\begin{verbatim}
[I=1,3 J=1,5] BX(I,J)
\end{verbatim}

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.

    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.

    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.

```
       15   4   50
```
(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.

    TYPE "THE VALUE OF THE PH IS"LOG(H)

Notice that no comma is needed between the second quotation mark and the mathematical expression 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

(a) \([f.d]\)
(b) \([An]\)

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 alphanumerical 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  26  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 TYPE statement. These characters can be suppressed by terminating the TYPE statement with a comma as shown in the following example.

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

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

5.3.6.2 The ATYPE Statement

The ATYPE statement is used to type out arrays of numbers. The form of the ATYPE statement is

\[
\text{ATYPE array}
\]

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.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5</td>
<td>6.56</td>
</tr>
<tr>
<td>2</td>
<td>7.8</td>
</tr>
<tr>
<td>9.3</td>
<td>500</td>
</tr>
</tbody>
</table>

5-33
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.

```
ATYPE AX(3,2)[8.2]
```

The following output might result from this statement.

```
2.50  6.56
2.00  7.80
9.30  500.00
```

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

```
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.

```
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

```
IF relation, statement
```

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

\[ \text{RESERVE } n \]

where \( n \) is the number of words being reserved. \( \text{RESER} \) may be used in place of \( \text{RESERVE} \).

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.

\[ \text{REM THIS FRAME GENERATES RANDOM DATA.} \]

5.3.12 Example C-frames

Example #1

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

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

**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 \( M \text{[1.0]} \text{KG} \) ACCELERATES AT THE RATE OF \( A \text{[1.0]} \text{M/SEC/SEC}. \) 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?

* 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
SEDIMENTARY ROCKS
```

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

```
TOPIC #3
SEDIMENTARY 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.
```
CHAPTER 6

RECORD KEEPING AND REPORT GENERATION

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.
CHAPTER 7

TESTING LEARN LESSONS

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!
```

7-2

FM. 60

THE FACT THAT * 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 0

FM. 120

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

AP, APH

TOTAL ERRORS DETECTED: 0

START AT FRAME 120

FM. 120

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

FINE!
FM. 130

PHANERITIC ROCKS FORM FROM MAGMA WHICH COOLS SLOWLY. PHANERITIC ROCKS ARE SAID TO BE:

ENTER COMMAND

*BYE
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

```
8-1
```
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

* S
CALCULATION MODE
*TYPE 24*15
-360
*TYPE EXP((3/4)*NLOG(16))
8
[I=1,10]TYPE I[5.0],SQRT(I)[9.3]

<table>
<thead>
<tr>
<th>I</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.000</td>
</tr>
<tr>
<td>2</td>
<td>1.414</td>
</tr>
<tr>
<td>3</td>
<td>1.732</td>
</tr>
<tr>
<td>4</td>
<td>2.000</td>
</tr>
<tr>
<td>5</td>
<td>2.236</td>
</tr>
<tr>
<td>6</td>
<td>2.449</td>
</tr>
<tr>
<td>7</td>
<td>2.646</td>
</tr>
<tr>
<td>8</td>
<td>2.828</td>
</tr>
<tr>
<td>9</td>
<td>3.000</td>
</tr>
<tr>
<td>10</td>
<td>3.162</td>
</tr>
</tbody>
</table>
* ARRAY AM(1) 
* 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
APPENDIX A

THE SLIDE PROJECTOR INTERFACE*

The slide projector interface may be used to control either a Kodak Ektagraphic 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.

A-1

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
RELIABLE 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.
```
APPENDIX D

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) TELETYPE
2) TEK 4010
3) TEK 4002
\ANSWER
1 ø 1
2 ø 2
3 ø 3
? ø
ACTIONS
1 2 3
C:$D=$A - 1
@R:PLEASE TYPE 1, 2, OR 3.\n```

F-1
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>

G-1
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

<table>
<thead>
<tr>
<th>A</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviated commands, 3-1</td>
<td>D command, 4-3</td>
</tr>
<tr>
<td>A command, 4-2</td>
<td>DELETE command, 3-12</td>
</tr>
<tr>
<td>action labels, 5-14</td>
<td>DELIM statement, 5-9</td>
</tr>
<tr>
<td>ACTION section, 5-2, 5-10</td>
<td>device numbers, 5-4, 5-13, 5-25, F-1</td>
</tr>
<tr>
<td>ACTION statement, 5-10</td>
<td>disk, 1-2</td>
</tr>
<tr>
<td>alphabetic labels, 5-7</td>
<td>E</td>
</tr>
<tr>
<td>answer comparison switches, 5-9</td>
<td>E command, 4-7</td>
</tr>
<tr>
<td>answer files, 2-3, 3-2, 6-2</td>
<td>EDIT command, 3-3, 4-1</td>
</tr>
<tr>
<td>ANSWER section, 5-2, 5-6</td>
<td>EP command, 4-7</td>
</tr>
<tr>
<td>ANSWER statement, 5-6</td>
<td>error limits, 5-8, 5-25</td>
</tr>
<tr>
<td>anticipated answers, 5-7</td>
<td>E: statement, 5-14</td>
</tr>
<tr>
<td>ARRAY statement, 5-29, 8-1</td>
<td>ET command, 4-7</td>
</tr>
<tr>
<td>array variables, 5-24</td>
<td>EXTRA statement, 5-9</td>
</tr>
<tr>
<td>assignment statement, 5-29, 8-1</td>
<td></td>
</tr>
<tr>
<td>A: statement, 5-11</td>
<td></td>
</tr>
<tr>
<td>ATIME function, 5-27</td>
<td></td>
</tr>
<tr>
<td>ATYPE statement, 5-33, 8-1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>F</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>B-frames, 5-1, 5-39</td>
<td>filename, D-1</td>
</tr>
<tr>
<td>BOTH statement, 5-6, 5-17</td>
<td>FILES command, 3-16</td>
</tr>
<tr>
<td>branching, 1-3</td>
<td>FILL statement, 5-31</td>
</tr>
<tr>
<td>B: statement, 5-12, 5-34</td>
<td>format specificiation, 5-3, 5-32</td>
</tr>
<tr>
<td>BYE command, 3-27</td>
<td>frame number, 2-1, 4-2, 5-1, 6-1</td>
</tr>
<tr>
<td></td>
<td>frame number increment, 3-2, 3-4, 4-2, 4-7</td>
</tr>
<tr>
<td></td>
<td>frame type, 2-1, 5-1</td>
</tr>
<tr>
<td></td>
<td>F: statement, 5-10</td>
</tr>
<tr>
<td></td>
<td>functions</td>
</tr>
<tr>
<td></td>
<td>mathematical, 5-26</td>
</tr>
<tr>
<td></td>
<td>special, 5-27</td>
</tr>
<tr>
<td>C</td>
<td>G</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>CALC statement, 5-5, 8-1</td>
<td>GET command, 2-4, 3-7</td>
</tr>
<tr>
<td>calculation mode, 1-3, 8-1</td>
<td>GOTO statement, 5-34</td>
</tr>
<tr>
<td>CALL statement, 5-35</td>
<td>G: statement, 5-12, 5-34</td>
</tr>
<tr>
<td>C command, 4-3</td>
<td></td>
</tr>
<tr>
<td>C-frames, 5-1, 5-21</td>
<td></td>
</tr>
<tr>
<td>character constants, 5-23</td>
<td></td>
</tr>
<tr>
<td>CLEAR command, 2-3, 3-20, 6-2</td>
<td></td>
</tr>
<tr>
<td>constants, 5-23</td>
<td></td>
</tr>
<tr>
<td>COPY command, 3-10</td>
<td></td>
</tr>
<tr>
<td>CORE statement, 8-1</td>
<td></td>
</tr>
<tr>
<td>CRT displays, 1-2, 5-4, 5-13</td>
<td></td>
</tr>
<tr>
<td>C: statement, 5-13</td>
<td></td>
</tr>
</tbody>
</table>

INDEX-1

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
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
interation 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
regular variables, 5-24
relational symbols, 5-35
R-frames, 5-1, 5-38
REMEM 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

T

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, l-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

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
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
THEM MAY BE RUN WITH EITHER A KAI0 OR K110 CPU RUNNING UNDER
THE CONTROL OF A 906 MONITOR. THE LEARN SYSTEM WAS DEVELOPED BY
GARRET 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 IS COMPOSED OF FOUR REENTRANT HIGH SEGMENTS - LEARN.SHR,
LEDIT.SHR, LRPRT.SHR, AND LPROS.SHR. THE FOUR HIGH SEGMENTS
CONTAIN THE FOUR MAJOR COMPONENTS OF THE LEARN SYSTEM - THE
LESSON INTERPRETER (LEARN.SHR), THE LESSON EDITOR (LEDIT.SHR),
THE REPORT GENERATOR (LRPRT.SHR), AND THE LESSON PROCESSOR
(LPROS.SHR).

LEARN.SHR IS COMPOSED OF THE FOLLOWING FILES:

LEARN.MAC
INTERP.MAC
CALC.MAC
PROS.MAC
FUNCS.MAC
ACTION.MAC
AUX1.MAC
LOW1.MAC (DUMMY LOW SEGMENT)

LEDIT.SHR IS COMPOSED OF THE FOLLOWING FILES:

LEDIT.MAC
PERL.MAC
LPRINT.MAC
WRITE.MAC
AUX2.MAC
LOW1.LOW (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
CFRAM.MAC
AUX4.MAC
PRSLOW.MAC (DUMMY LOW SEGMENT)

THE FILES FOR EACH HIGH SEGMENT SHOULD BE LOADED IN THE ORDER ShOWN ABOVE AND SAVED USING THE SSAVE 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. VANDER LUST 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 888888

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. CPROM.MAC
5. FUNCS.MAC
6. ACTION.MAC
7. AUX1.MAC
8. L0W1.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. KJOB
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
REPLACES ITSELF WITH THE HIGH SEGMENT WHICH CAN EXECUTE THE COMMAND.
WHEN LEARN.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. LEARN.MAC ALSO
DOES LOOKUPS FOR LESSONS SPECIFIED IN LEARN "GET" COMMANDS.

CONTENTS OF LEARN.MAC

SECTION A. PERFORMS THE GETSEG UUD.
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 LPRT HIGH SEGMENT.
SECTION L. CONTINUES THE EXECUTION OF A TEST COMMAND.
SECTION M. INPUTS THE LESSON NAME IN A "GET" COMMAND.
SECTION N. INPUTS PROG-NUMBERS IN A "GET" COMMAND.
SECTION O. DOES A LOOKUP FOR THE LESSON SPECIFIED IN A "GET" COMMAND.
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 "LI,
LOCATED IN SECTION P

CHD - FROM INTERP.MAC (SECTION WH) AT THE CONCLUSION OF A
LESSON, LOCATED IN SECTION E
GETSEG = FROM INTERP,MAC (SECTION C) 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)

EOLIN = INPUTS CHARACTERS FROM A USER'S TERMINAL UNIL 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 WO (6 WORDS)

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

IBUF = INPUT BUFFER HEADER (3 WORDS)

INAM = LOOKUP ARGUMENTS (4 WORDS)

INBUF = DISK INPUT BUFFER (33 WORDS)

INTBK = CONTROL C INTERRUPT BLOCK (5 WORDS)

INTLOC = CONTROL C INTERRUPT ROUTINE (14 WORDS)

JOBNO = USER'S JOB NUMBER

LKSAY = 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
DESCRIPTION OF FLAGS STORED IN AO 9

BIT 0 TEST LESSON
BIT 1 INFORMATION LESSON

MISEG

INTERNAL GETSG,BYE, CMD

EXTERN ARGMT,CALC,M, EOLIN, IBUF, INAH, INBUF, INTBLK, INTERP, INCOL, JOBNO,
EXTERN IBUF, ONAH, OUTBUF, PRN, RSNUM, STORE, TRMBLK, LKS, DATA, ADDR, SYMIDX

000000
000001
000002
000003
000004
000005
000006
000007
000017

FLAGS=8
A=1
B=2
C=3
D=4
E=5
F=6
G=7

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

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

000001 84 01 0 0 000000
GETSG1 GETSEG A;

MALT

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.
<table>
<thead>
<tr>
<th>LEARN</th>
<th>MACRO</th>
<th>47(204)-7 08105</th>
<th>2-AUG-74 PAGE 1-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0213</td>
<td>00001</td>
<td>MOVE B,44</td>
<td>GET HIGHEST LOWSEG ADDRESS</td>
</tr>
<tr>
<td>0214</td>
<td>000031</td>
<td>SUBI B,DATA</td>
<td>COMPUTE NO. OF WORDS OF FREE STORE</td>
</tr>
<tr>
<td>0215</td>
<td>000041</td>
<td>MOVEN B,RSVNUM</td>
<td>RESERVE CORE FOR CALC MODE DATA</td>
</tr>
<tr>
<td>0216</td>
<td>000051</td>
<td>SETSYM SYNDX</td>
<td>SET SYMBOL TABLE INDEX TO ZERO</td>
</tr>
<tr>
<td>0217</td>
<td>000061</td>
<td>MOVAR A,422</td>
<td>CLEAR THE ADDRES REGISTER</td>
</tr>
<tr>
<td>0218</td>
<td>000071</td>
<td>MOVER B,STOR</td>
<td>GET THE COMMAND INDEX IN AC-8</td>
</tr>
<tr>
<td>0219</td>
<td>000081</td>
<td>JRST 0,(B)</td>
<td></td>
</tr>
<tr>
<td>0220</td>
<td>000091</td>
<td>XW0 G,TEST2</td>
<td>SEE SECTION L</td>
</tr>
<tr>
<td>0221</td>
<td>000111</td>
<td>XW0 G,GET</td>
<td>SEE SECTION M</td>
</tr>
<tr>
<td>0222</td>
<td>000131</td>
<td>XW0 D,CALC</td>
<td>SEE SECTION I</td>
</tr>
<tr>
<td>0223</td>
<td>000141</td>
<td>XW0 D,SEP</td>
<td>SEE SECTION F</td>
</tr>
<tr>
<td>0224</td>
<td>000001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0225</td>
<td>000011</td>
<td></td>
<td>SECTION C. INITIALIZE THE PROGRAM. SAVE ARGUMENTS FOR GETSEG UVO</td>
</tr>
<tr>
<td>0226</td>
<td>000021</td>
<td></td>
<td>WHICH HAVE BEEN PLACED IN ACCUMULATORS D, 7, AND 11 BY THE MONITOR.</td>
</tr>
<tr>
<td>0227</td>
<td>000031</td>
<td></td>
<td>OPEN CHANNELS 1 AND 2, AND ESTABLISH BUFFERS. TRANSFER A ROUTINE</td>
</tr>
<tr>
<td>0228</td>
<td>000041</td>
<td></td>
<td>FOR HANDLING CONTROL C'S TO THE LOW SEGMENT.</td>
</tr>
<tr>
<td>0229</td>
<td>000051</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0230</td>
<td>000011</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0231</td>
<td>000011</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0232</td>
<td>000011</td>
<td>00 07 0 0 000080</td>
<td>POINT1 POINT 7,0,35</td>
</tr>
<tr>
<td>0233</td>
<td>000011</td>
<td>000000 000000</td>
<td>OPN1 0</td>
</tr>
<tr>
<td>0234</td>
<td>000011</td>
<td>44 63 53 00 00</td>
<td>SIXBIT /DSDK/</td>
</tr>
<tr>
<td>0235</td>
<td>000011</td>
<td>000000 000000</td>
<td>XW0 G,IBUF</td>
</tr>
<tr>
<td>0236</td>
<td>000011</td>
<td>000000 000000</td>
<td>OPN2 0</td>
</tr>
<tr>
<td>0237</td>
<td>000011</td>
<td>44 63 53 00 00</td>
<td>SIXBIT /DSDK/</td>
</tr>
<tr>
<td>0238</td>
<td>000011</td>
<td>000000 000000</td>
<td>XW0 D,IBUF 0</td>
</tr>
<tr>
<td>0239</td>
<td>000011</td>
<td>406 06 0 0 000000</td>
<td>START1 SETBC F,ADDR .STARTING ADDRESS FOR LEARN</td>
</tr>
<tr>
<td>0240</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVEM 6,ARGMT+1 ISAVE PROGRAM NAME IN ARGMNT+1</td>
</tr>
<tr>
<td>0241</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVEM 7,ARGMT+4 ISAVE PROM-PROG FOR GETSEG ARGUMENT</td>
</tr>
<tr>
<td>0242</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVEM 11,ARGMT+6 ISAVE THE DEVICE NAME FOR GETSEG ARGUMENT</td>
</tr>
<tr>
<td>0243</td>
<td>000011</td>
<td>402 00 0 0 000000</td>
<td>SETEM ARGMT+2</td>
</tr>
<tr>
<td>0244</td>
<td>000011</td>
<td>402 00 0 0 000000</td>
<td>SETEM ARGMT+3</td>
</tr>
<tr>
<td>0245</td>
<td>000011</td>
<td>402 00 0 0 000000</td>
<td>SETEM ARGMT+5</td>
</tr>
<tr>
<td>0246</td>
<td>000011</td>
<td>402 00 0 0 000000</td>
<td>JRST 1+3</td>
</tr>
<tr>
<td>0247</td>
<td>000011</td>
<td>000000 000000</td>
<td>RESTRT1 SETO 8</td>
</tr>
<tr>
<td>0248</td>
<td>000011</td>
<td>000000 000000</td>
<td>RETCALL 3,CASCII /</td>
</tr>
<tr>
<td>0249</td>
<td>000011</td>
<td>000000 000000</td>
<td>TYPE &quot;HON&quot; TO EXIT FROM LEARN</td>
</tr>
<tr>
<td>0250</td>
<td>000011</td>
<td>000000 000000</td>
<td>/</td>
</tr>
<tr>
<td>0251</td>
<td>000011</td>
<td>000000 000000</td>
<td>RESET</td>
</tr>
<tr>
<td>0252</td>
<td>000011</td>
<td>000000 000000</td>
<td>OPEN 1,0PN1 IASCII MODE</td>
</tr>
<tr>
<td>0253</td>
<td>000011</td>
<td>000000 000000</td>
<td>HALT</td>
</tr>
<tr>
<td>0254</td>
<td>000011</td>
<td>000000 000000</td>
<td>OPEN 2,0PN2 IASCII MODE</td>
</tr>
<tr>
<td>0255</td>
<td>000011</td>
<td>000000 000000</td>
<td>HALT</td>
</tr>
<tr>
<td>0256</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,[XW0 400062,CUTBUF+13]</td>
</tr>
<tr>
<td>0257</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,IBUF ISET UP OUTPUT BUFFER HEADER</td>
</tr>
<tr>
<td>0258</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,[XW0 400092,CUTBUF+13]</td>
</tr>
<tr>
<td>0259</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,IBUF ISET UP INPUT BUFFER HEADER</td>
</tr>
<tr>
<td>0260</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,POIN</td>
</tr>
<tr>
<td>0261</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,POIN</td>
</tr>
<tr>
<td>0262</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,IBUF+1</td>
</tr>
<tr>
<td>0263</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,[XW0 251,INBUF+13]</td>
</tr>
<tr>
<td>0264</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,INBUF+1 ISET UP INPUT BUFFER</td>
</tr>
<tr>
<td>0265</td>
<td>000011</td>
<td>000000 000000</td>
<td>MOVE A,[XW0 251,CUTBUF+13]</td>
</tr>
<tr>
<td>LEARN MACRO 47(204)-7 02105 2-AUG-74 PAGE 1-6</td>
<td>LEARN MAC 8-JUN-74</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>319 00132' 001 03 00 0000566'</td>
<td>GOOD AFTERNOON /3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>320 00130' 254 00 00 000132'</td>
<td>JRS Y MESS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>321 00111' 254 00 00 0000572'</td>
<td>MORN, TDCALL 3, 3ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td>322 00131' 051 03 00 0000572'</td>
<td>GOOD MORNIG! /3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>323 00132' 001 03 00 0000566'</td>
<td>MESSI TDCALL 3, 3ASCII /THIS IS LEARN (301A).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>324</td>
<td>TYPE H FOR HELP.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>325 00132' 001 03 00 0000576'</td>
<td>MOVE B,C(SIXBIT/INFO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>326 00139' 200 02 00 0000607'</td>
<td>MOVE ADDRESS OF TRMOP, ARGUMENTS TO ACC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>327 00134' 202 02 00 0000570'</td>
<td>MOVE ADDRESS (IN AM) TO LOOKUP ARGUMENTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>328 00136' 661 02 00 0000570'</td>
<td>TLO FLAGS:200000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>329 00136' 621 00 00 00000000</td>
<td>TLO FLAGS:400000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>330 00137' 254 00 00 0000605'</td>
<td>JRT GETINF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>331</td>
<td>ISEE SECTION 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>332</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>333</td>
<td>SECTION E. INPUT THE FIRST WORD IN A LEARN SYSTEM COMMAND.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>334</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>335 00140' 200 01 00 0000646'</td>
<td>CMDI MOVE A,TRMPLK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>336 00141' 047 01 00 0000156</td>
<td>TRMOP, A,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>337 00142' 201 02 00 0000145</td>
<td>JRT IGNIP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>338 00143' 201 02 00 0000145</td>
<td>MOVE B,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>339 00144' 047 02 00 0000931</td>
<td>SLEEP,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>340 00145' 204 00 00 000141</td>
<td>JRT -1,4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>341 00146' 051 14 00 00000000</td>
<td>EMPTY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>342 00147' 255 00 00 00000000</td>
<td>TDCALL 14,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>343</td>
<td>JOVERID USER'S CONTROL O, IF ANY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>344</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>345 00150' 051 23 00 0000618'</td>
<td>TDCALL 3, 3ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td>346 00151' 403 22 00 00002004</td>
<td>ENTER COMMAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>347 00152' 200 02 00 0000261</td>
<td>SETEB O,D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>348 00153' 084 03 00 0000058</td>
<td>MOVE TO POINT 6:03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>349 00154' 093 06 00 0000050</td>
<td>SETO E,1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>350 00155' 200 21 00 0000040</td>
<td>TDCALL 4,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>351 00156' 025 00 00 0000154</td>
<td>SET UP CHARACTER DELETION MASK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>352 00157' 304 21 00 00000000</td>
<td>TDCALL 4,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>353 00158' 304 21 00 00000000</td>
<td>INPUT CHARACTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>354 00159' 200 21 00 0000030</td>
<td>CAIN A,40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>355 00156' 254 00 00 0000154</td>
<td>IS CHARACTER A SPACE?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>356 00157' 304 21 00 00000000</td>
<td>JRT -2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>357 00160' 304 21 00 00000000</td>
<td>YES, GO IN THE NEXT CHARACTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>358 00159' 304 21 00 00000000</td>
<td>CAI A,33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>359 00159' 304 21 00 00000000</td>
<td>CAIN A,40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>360 00160' 254 00 00 0000154</td>
<td>DID USER TYPE &quot;ESC&quot;?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>361 00161' 304 21 00 00000000</td>
<td>JRT CALCI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>362 00162' 304 21 00 00000000</td>
<td>YES, GO INTO CALC MODE (SECTION M)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>363 00163' 304 21 00 00000000</td>
<td>CAI A,15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>364 00164' 254 00 00 0000154</td>
<td>DID USER TYPE A CARRIAGE RETURN?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>365 00165' 304 21 00 00000000</td>
<td>JRT CHD2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>366 00166' 304 21 00 00000000</td>
<td>IND, GO TO INPUT COMMAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>367 00167' 304 21 00 00000000</td>
<td>TDCALL 4,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>368 00168' 304 21 00 00000000</td>
<td>INPUT LINE FEED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>369 00169' 254 20 00 0000140</td>
<td>JRT CHD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>370 00166' 051 04 00 0000201</td>
<td>CMDM TDCALL 4,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>371 00167' 302 21 00 0000240</td>
<td>INPUT NEXT CHARACTER IN COMMAND STRING</td>
<td></td>
<td></td>
</tr>
<tr>
<td>372 00168' 302 21 00 0000240</td>
<td>CAI A,40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>373 00169' 302 21 00 0000240</td>
<td>CAIN A,15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>374 00170' 254 20 00 0000231</td>
<td>JRT CHDIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>375 00171' 254 20 00 0000231</td>
<td>IF END OF WORD, GO TO CHDIN (SECTION F)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>376 00172' 306 21 00 0000057</td>
<td>JRT CHDIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>377 00173' 254 20 00 0000231</td>
<td>IF END OF WORD, GO TO CHDIN (SECTION F)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>378 00174' 301 21 00 0000121</td>
<td>CMD21 CALL A,&quot;A&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>379 00175' 303 21 00 0000121</td>
<td>JRT CHDERR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>380 00176' 254 20 00 0000205</td>
<td>IF CHARACTER IS NOT ALPHABETIC, GO TYPE ERROR MESSAGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>381 00177' 306 21 00 0000057</td>
<td>CALL A,15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>382 00178' 306 21 00 0000057</td>
<td>I HAVE MORE THAN 6 CHARACTERS BEEN TYPED?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>383 00179' 254 20 00 0000231</td>
<td>JRT CHD2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>384 00180' 301 21 00 0000121</td>
<td>YES, IGNORE THIS CHARACTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>385 00181' 254 20 00 00000000</td>
<td>ADDI A,40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>386 00182' 254 20 00 00000000</td>
<td>I CONVET CHARACTERS TO SIXBIT CODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>387 00183' 136 01 00 000002</td>
<td>IDPB A,B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>388 00184' 136 01 00 000002</td>
<td>IPLACE CHARACTER IN AC-D</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECTION F. INTERPRET THE FIRST WORD IN A LEARN SYSTEM COMMAND.
<table>
<thead>
<tr>
<th>LEARN MACRO</th>
<th>47(204)-7 02/05 2-AUG-74 PAGE 1-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>425 002241 002002 000337</td>
<td>XWD B,DELETE ISEE SECTION J</td>
</tr>
<tr>
<td>426 002262 000000 000317</td>
<td>XWD B,FILES ISEE SECTION I</td>
</tr>
<tr>
<td>427 002251 000000 000332</td>
<td>XWD B,KDB ISEE SECTION G</td>
</tr>
<tr>
<td>428 002264 000250 000301</td>
<td>XWD B,MONIT ISEE SECTION G</td>
</tr>
<tr>
<td>429 002269 000800 002866</td>
<td>XWD B,HELP</td>
</tr>
<tr>
<td>430 002246 269 17 0 00 002002, HELP I INPUT REST OF COMMAND STRING</td>
<td></td>
</tr>
<tr>
<td>431 002269 200 02 00 002002</td>
<td>MOVE B,(SIXBITHELP)</td>
</tr>
<tr>
<td>432 002270 200 02 00 001344</td>
<td>MOVE B,INAM IPUT THEHELP IN INAM</td>
</tr>
<tr>
<td>433 002274 200 02 00 002004</td>
<td>MOVE B,ARCHNT-4 IGET LEARN,SRP,PPN</td>
</tr>
<tr>
<td>434 002272 200 02 00 000993</td>
<td>MOVE B,INAM+3 IPUT IT IN INAM+3</td>
</tr>
<tr>
<td>435 002273 641 02 00 002000</td>
<td>FILE FLAGS,200320 ISET INFORMATION FILE FLAG</td>
</tr>
<tr>
<td>436 002274 621 00 00 000000</td>
<td>FILE FLAGS,400205 CLEAR TEST MODE FLAG</td>
</tr>
<tr>
<td>437 002275 254 02 00 000478</td>
<td>JST GETL IGO LOOKUP HELP LESSON (SECTION O)</td>
</tr>
<tr>
<td>438 002276 051 11 00 000000</td>
<td>CMER21 TCALL 11,8</td>
</tr>
<tr>
<td>439 002277 051 13 00 000452</td>
<td>AMBIGUOUS ABBREVIATION, TRY AGAIN</td>
</tr>
<tr>
<td>440 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>441 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>442 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>443 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>444 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>445 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>446 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>447 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>448 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>449 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>450 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>451 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>452 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>453 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>454 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>455 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>456 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>457 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>458 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>459 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>460 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>461 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>462 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>463 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>464 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>465 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>466 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>467 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>468 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>469 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>470 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>471 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>472 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>473 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>474 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>475 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>476 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>477 002301 254.00 00 000140</td>
<td>JST CHD ISEE SECTION C</td>
</tr>
<tr>
<td>LRN</td>
<td>MAC</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>531</td>
<td>022372</td>
</tr>
<tr>
<td>532</td>
<td>022373</td>
</tr>
<tr>
<td>533</td>
<td>022374</td>
</tr>
<tr>
<td>534</td>
<td>022375</td>
</tr>
<tr>
<td>535</td>
<td>022376</td>
</tr>
<tr>
<td>536</td>
<td>022377</td>
</tr>
<tr>
<td>537</td>
<td>004040</td>
</tr>
<tr>
<td>538</td>
<td>004041</td>
</tr>
<tr>
<td>539</td>
<td>004042</td>
</tr>
<tr>
<td>540</td>
<td></td>
</tr>
<tr>
<td>541</td>
<td></td>
</tr>
<tr>
<td>542</td>
<td>004043</td>
</tr>
<tr>
<td>543</td>
<td>004044</td>
</tr>
<tr>
<td>544</td>
<td>004045</td>
</tr>
<tr>
<td>545</td>
<td></td>
</tr>
<tr>
<td>546</td>
<td></td>
</tr>
<tr>
<td>547</td>
<td>004046</td>
</tr>
<tr>
<td>548</td>
<td>004047</td>
</tr>
<tr>
<td>549</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td>004048</td>
</tr>
<tr>
<td>554</td>
<td>004049</td>
</tr>
<tr>
<td>555</td>
<td>004050</td>
</tr>
<tr>
<td>556</td>
<td>004051</td>
</tr>
<tr>
<td>557</td>
<td>004052</td>
</tr>
<tr>
<td>558</td>
<td>004053</td>
</tr>
<tr>
<td>559</td>
<td>004054</td>
</tr>
<tr>
<td>560</td>
<td>004055</td>
</tr>
<tr>
<td>561</td>
<td>004056</td>
</tr>
<tr>
<td>562</td>
<td>004057</td>
</tr>
<tr>
<td>563</td>
<td>004058</td>
</tr>
<tr>
<td>564</td>
<td>004059</td>
</tr>
<tr>
<td>565</td>
<td>004060</td>
</tr>
<tr>
<td>566</td>
<td>004061</td>
</tr>
<tr>
<td>567</td>
<td>004062</td>
</tr>
<tr>
<td>568</td>
<td>004063</td>
</tr>
<tr>
<td>569</td>
<td>004064</td>
</tr>
<tr>
<td>570</td>
<td>004065</td>
</tr>
<tr>
<td>571</td>
<td>004066</td>
</tr>
<tr>
<td>572</td>
<td>004067</td>
</tr>
<tr>
<td>573</td>
<td>004068</td>
</tr>
<tr>
<td>574</td>
<td>004069</td>
</tr>
<tr>
<td>575</td>
<td>004070</td>
</tr>
<tr>
<td>576</td>
<td>004071</td>
</tr>
<tr>
<td>577</td>
<td>004072</td>
</tr>
<tr>
<td>578</td>
<td>004073</td>
</tr>
<tr>
<td>579</td>
<td>004074</td>
</tr>
<tr>
<td>580</td>
<td>004075</td>
</tr>
<tr>
<td>581</td>
<td>004076</td>
</tr>
<tr>
<td>582</td>
<td>004077</td>
</tr>
<tr>
<td>583</td>
<td>004078</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LEARN MAC</th>
<th>5-JUN-74</th>
</tr>
</thead>
<tbody>
<tr>
<td>531</td>
<td>JST CHK1</td>
</tr>
<tr>
<td>532</td>
<td>YES, GO TO CHK1 (SECTION N)</td>
</tr>
<tr>
<td>533</td>
<td>CAIN A,15</td>
</tr>
<tr>
<td>534</td>
<td>END OF COMMAND STRING?</td>
</tr>
<tr>
<td>535</td>
<td>JST GETLES</td>
</tr>
<tr>
<td>536</td>
<td>YES, GO TO GETLES (SECTION D)</td>
</tr>
<tr>
<td>537</td>
<td>CAIN C,8</td>
</tr>
<tr>
<td>538</td>
<td>MORE THAN 8 CHARACTERS IN LESSON NAME?</td>
</tr>
<tr>
<td>539</td>
<td>YES, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>540</td>
<td>ADDI A,49</td>
</tr>
<tr>
<td>541</td>
<td>CONVERT CHARACTER TO SIXBIT-CODE</td>
</tr>
<tr>
<td>542</td>
<td>IDPS A,8</td>
</tr>
<tr>
<td>543</td>
<td>MOVE CHARACTER TO INAM</td>
</tr>
<tr>
<td>544</td>
<td>ADJA Q, NAMIN</td>
</tr>
<tr>
<td>545</td>
<td>INCREMENT CHARACTER COUNT AND GO TO NAMIN</td>
</tr>
<tr>
<td>546</td>
<td></td>
</tr>
<tr>
<td>547</td>
<td></td>
</tr>
<tr>
<td>548</td>
<td></td>
</tr>
<tr>
<td>549</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td></td>
</tr>
<tr>
<td>554</td>
<td></td>
</tr>
<tr>
<td>555</td>
<td></td>
</tr>
<tr>
<td>556</td>
<td></td>
</tr>
<tr>
<td>557</td>
<td></td>
</tr>
<tr>
<td>558</td>
<td></td>
</tr>
<tr>
<td>559</td>
<td></td>
</tr>
<tr>
<td>560</td>
<td></td>
</tr>
<tr>
<td>561</td>
<td></td>
</tr>
<tr>
<td>562</td>
<td></td>
</tr>
<tr>
<td>563</td>
<td></td>
</tr>
<tr>
<td>564</td>
<td></td>
</tr>
<tr>
<td>565</td>
<td></td>
</tr>
<tr>
<td>566</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td></td>
</tr>
<tr>
<td>568</td>
<td></td>
</tr>
<tr>
<td>569</td>
<td></td>
</tr>
<tr>
<td>570</td>
<td></td>
</tr>
<tr>
<td>571</td>
<td></td>
</tr>
<tr>
<td>572</td>
<td></td>
</tr>
<tr>
<td>573</td>
<td></td>
</tr>
<tr>
<td>574</td>
<td></td>
</tr>
<tr>
<td>575</td>
<td></td>
</tr>
<tr>
<td>576</td>
<td></td>
</tr>
<tr>
<td>577</td>
<td></td>
</tr>
<tr>
<td>578</td>
<td></td>
</tr>
<tr>
<td>579</td>
<td></td>
</tr>
<tr>
<td>580</td>
<td></td>
</tr>
<tr>
<td>581</td>
<td></td>
</tr>
<tr>
<td>582</td>
<td></td>
</tr>
<tr>
<td>583</td>
<td></td>
</tr>
</tbody>
</table>

SECTION N. INPUT PPIN IN GET COMMAND IF THEY ARE PRESENT.
LEARN  MACRO 47 (204) = 08085  2-AUG-74 PAGE 1-12

637  00514  000 00 00 000000  BYE1
638  00515  001 11 00 000000
639  00516  201 01 00 000000
640  00517  247 81 00 000000
641  00518  294 04 00 000000

TCALL 11,0
MOVE A, RUNARG
PUT ADDRESS OF RUN ARGUMENTS IN AC-A
RUN A1
RUN THE LOGOUT SYSTEM PROGRAM
HALT

SECTION G. EXECUTE THE KJOB COMMAND BY RUNNING THE KJOB PROGRAM

067  00523  03 71 63 00 00 00  RARG  SIXBIT /SYS/
068  00524  53 52 57 42 00 00  SIXBIT /KJOB/
069  00525  00 00 07 00 000000
070  00526  00 00 07 00 000000
071  00527  00 00 07 00 000000
072  00528  00 00 07 00 000000
073  00529  00 00 07 00 000000
074  00530  00 00 07 00 000000
075  00531  00 00 07 00 000000
076  00532  00 00 07 00 000000
077  00533  00 00 07 00 000000
078  00534  00 00 07 00 000000
079  00535  00 00 07 00 000000
080  00536  00 00 07 00 000000
081  00537  00 00 07 00 000000
082  00538  00 00 07 00 000000
083  00539  00 00 07 00 000000

 floors

NOSYM

0863  00540  00 00 07 00 000000
0864  00541  00 00 07 00 000000
0865  00542  00 00 07 00 000000
0866  00543  00 00 07 00 000000

0867  00544  00 00 07 00 000000
0868  00545  00 00 07 00 000000

ISPECIFIES THE STARTING ADDRESS FOR THE LEARN SYSTEM
END START

NO ERRORS DETECTED

PROGRAM BREAK IS 000795

3K CORE USED
<table>
<thead>
<tr>
<th>ILLPN</th>
<th>560</th>
<th>577</th>
<th>581#</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>182#</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>303</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTBLK</td>
<td>182#</td>
<td>298</td>
<td>298</td>
<td>303</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTIP</td>
<td>182#</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>INLOC</td>
<td>182#</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>INTRTN</td>
<td>285#</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></td>
<td>277</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>654#</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>475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEAVE2</td>
<td>482#</td>
<td>488</td>
<td>490</td>
<td>492</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>499#</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>
</tr>
<tr>
<td>LIST</td>
<td>418</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>
</tr>
<tr>
<td>LKSAV</td>
<td>182#</td>
<td>639</td>
<td></td>
<td>640</td>
<td>323#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MESS</td>
<td></td>
<td>317</td>
<td>329</td>
<td>323#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HONIT</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></td>
<td>321#</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>536</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>NAMIN</td>
<td>527#</td>
<td></td>
<td>538</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NODANCE</td>
<td></td>
<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>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>182#</td>
<td></td>
<td>238</td>
<td>257</td>
<td>261</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBAM</td>
<td>182#</td>
<td></td>
<td>238</td>
<td>257</td>
<td>261</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPN1</td>
<td>231#</td>
<td></td>
<td>252</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPN2</td>
<td>236#</td>
<td></td>
<td>252</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>182#</td>
<td></td>
<td>252</td>
<td>265</td>
<td>266</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DINT</td>
<td></td>
<td>252</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>DINT</td>
<td></td>
<td>252</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>PPN1</td>
<td>571</td>
<td></td>
<td></td>
<td>592</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></td>
<td></td>
<td>592</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRSS</td>
<td>416</td>
<td></td>
<td>504#</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>193#</td>
<td></td>
<td>438</td>
<td>447</td>
<td>453</td>
<td>596</td>
<td>603</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAG</td>
<td></td>
<td>414</td>
<td></td>
<td>472#</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>420</td>
<td></td>
<td>472#</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></td>
<td>481#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESTART</td>
<td></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>182#</td>
<td></td>
<td></td>
<td>298</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUNARGS</td>
<td></td>
<td>218</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>SETINT</td>
<td></td>
<td></td>
<td></td>
<td>309#</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></td>
<td>236#</td>
<td>668</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></td>
<td></td>
<td></td>
<td>462</td>
<td>483</td>
<td>500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYMDF</td>
<td></td>
<td></td>
<td></td>
<td>462</td>
<td>483</td>
<td>500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST</td>
<td></td>
<td>416</td>
<td></td>
<td></td>
<td>498#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>498#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRM0BLK</td>
<td></td>
<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>UPDATE</td>
<td></td>
<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>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>
<tr>
<td>Line</td>
<td>Description</td>
<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>------</td>
<td>-------------</td>
<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>TITLE INTERP</td>
<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>COMMENT X</td>
<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>WRITTEN BY GARRET A. VANDER LUST COMPLETED 3-MAY-74</td>
<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>PART OF LEARN.SHR, VERSION 3</td>
<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>INTERP.MAC INTERPRETS PROCESSED LESSON FILES. IN PARTICULAR: IT INTERPRETS G-FRAMES, R-FRAMES, AND B-FRAMES. C-FRAMES ARE INTERPRETED BY CALC.MAC. INTERP.MAC ALSO STORES STUDENT PERFORMANCE DATA.</td>
<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>NOTE: SEE PROS.MAC AND GFRAM.MAC FOR A LIST OF THE STATEMENT CODES WHICH MAY APPEAR IN A PROCESSED LESSON FILE.</td>
<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>CONTENTS OF INTERP.MAC</td>
<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>SECTION A. INITIALIZES THE LESSON INTERPRETER.</td>
<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>SECTION B. INTERPRETS THE FIRST LINE IN A FRAME.</td>
<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>SECTION C. INTERPRETS THE FIRST CHARACTER IN A G-FRAME STATEMENT.</td>
<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>SECTION D. SETS CALC, HINT, BOTH, STORE, REVIEW, AND RECORD FLAGS.</td>
<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>SECTION E. TYPES THE TEXT IN A TEXT STATEMENT.</td>
<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>SECTION F. INTERPRETS SLIDE STATEMENTS AND PAGES CRT TERMINALS.</td>
<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>SECTION G. ACCEPTS STUDENT ANSWERS.</td>
<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>SECTION H. PROCESSES HINT REQUESTS, REVIEW REQUESTS, AND CALC MODE REQUESTS.</td>
<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>SECTION I. INTERPRETS ANSWER TAGS, ANSWER EVALUATION MODE STATEMENTS, AND RIGHT-WRONG-NEUTRAL INDICATORS.</td>
<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>SECTION J. CONVERTS STUDENT ANSWERS TO FLOATING POINT NOS.</td>
<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>SECTION K. INTERPRETS CALC STATEMENTS AFTER &quot;G&quot; TAGS.</td>
<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>SECTION L. SETS EXTRA, ORDER, AND PHONIC FLAGS.</td>
<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>SECTION M. INTERPRETS MATCH STATEMENTS.</td>
<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>21</td>
<td>SECTION N. INTERPRETS DELIM STATEMENTS.</td>
<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>22</td>
<td>SECTION O. SETS UP LIST OF STANDARD WORD DELIMITERS.</td>
<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>23</td>
<td>SECTION P. SELECTS PROPER ANSWER EVALUATION MODE.</td>
<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>
SECTION O. 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

ANSH - FROM ACTION.MAC (SECTION D), LOCATED IN SECTION G.

EOH - FROM CALC.MAC (SECTION A) AND ACTION.MAC (SECTIONS H & J),
LOCATED IN SECTION HH.

FS1IN - FROM CALC.MAC (SECTIONS A, P & D) AND ACTION.MAC
(SECTIONS J & K), LOCATED IN SECTION B.

GETAC - FROM ACTION.MAC (SECTIONS A, C, E, G & L), LOCATED
IN SECTION E.

INTERP - FROM LEARN.MAC (SECTION D), LOCATED IN SECTION A.
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>LOGOUT - FROM ACTION.MAC (SECTION 1), LOCATED IN SECTION HM</td>
</tr>
<tr>
<td>108</td>
<td>PAG - FROM ACTION.MAC (SECTION 4), LOCATED IN SECTION F</td>
</tr>
<tr>
<td>109</td>
<td>QFRAME - FROM ACTION.MAC (SECTION F), LOCATED IN SECTION C</td>
</tr>
<tr>
<td>110</td>
<td>SKPTXT - FROM CALC.MAC (SECTION P) AND ACTION.MAC (SECTION J), LOCATED IN SECTION AA</td>
</tr>
<tr>
<td>111</td>
<td>SLDRTN - FROM ACTION.MAC (SECTION G), LOCATED IN SECTION F</td>
</tr>
<tr>
<td>112</td>
<td>UNCALL - FROM ACTION.MAC (SECTIONS A, C, G &amp; L), LOCATED IN SECTION 2</td>
</tr>
</tbody>
</table>

**EXTERNAL SUBROUTINES USED BY INTERP.MAC**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>ACHO - TYPES THE CORRECT ANSWERS, LOCATED IN ACTION.MAC (SECTION E)</td>
</tr>
<tr>
<td>123</td>
<td>BCHO - EXECUTES BRANCHING (BI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION K)</td>
</tr>
<tr>
<td>124</td>
<td>BCEHD2 - EXECUTES BRANCHING, LOCATED IN ACTION.MAC (SECTION K)</td>
</tr>
<tr>
<td>125</td>
<td>BCEHD2 - EXECUTES BRANCHING, LOCATED IN ACTION.MAC (SECTION K)</td>
</tr>
<tr>
<td>126</td>
<td>BYE - RUNS THE LOGOUT PROGRAM, LOCATED IN LEARN.MAC (SECTION P)</td>
</tr>
<tr>
<td>127</td>
<td>CALC - ENTRY POINT TO THE CALCULATION MODE, LOCATED IN CALC.MAC (SECTION A)</td>
</tr>
<tr>
<td>128</td>
<td>CDCM - EXECUTES CALC (CI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION C)</td>
</tr>
<tr>
<td>129</td>
<td>CFRAME - EXECUTES C-FRAME STATEMENTS, LOCATED IN CALC.MAC (SECTION A)</td>
</tr>
<tr>
<td>130</td>
<td>ECHO - EXECUTES EXIT (EI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION H)</td>
</tr>
<tr>
<td>131</td>
<td>EXPCOR - EXPANDS THE LOW SEGMENT IF NECESSARY, LOCATED IN AUX1.MAC (SECTION H)</td>
</tr>
<tr>
<td>132</td>
<td>FCHD - EXECUTES FEEDBACK (FI) STATEMENTS, LOCATED IN ACTION.MAC (SECTION A)</td>
</tr>
<tr>
<td>133</td>
<td>FINL - CONVERTS THE NO. IN ACM TO AN INTEGER, LOCATED IN FUNCS.MAC (SECTION B)</td>
</tr>
<tr>
<td>134</td>
<td>GCMD - EXECUTES SUBROUTINE CALLS (GI), LOCATED IN ACTION.MAC (SECTION J)</td>
</tr>
<tr>
<td>135</td>
<td>GETSG - EXECUTES THE GETSEG UUD, LOCATED IN LEARN.MAC (SECTION A)</td>
</tr>
</tbody>
</table>
IID = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1.MAC (SECTION E)

IID2 = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1.MAC (SECTION E)

INFLP = "INPUTS" A FLOATING POINT NO. INTO AC-B, LOCATED IN AUX1.MAC (SECTION F)

IOT = OUTPUTS THE INTEGER IN AC-A TO THE USER'S TERMINAL, LOCATED IN AUX1.MAC (SECTION G)

LCMD = EXECUTES LEAVE (LI) STATEMENTS, LOCATED IN AUX1.MAC (SECTION I)

OUTCHR = DISK OUTPUT ROUTINE, LOCATED IN AUX1.MAC (SECTION G)

OUTFLP = OUTPUTS THE FLOATING POINT NO. IN AC-A, LOCATED IN AUX1.MAC (SECTION G)

PCHD = EXECUTES PAGE (P1) STATEMENTS, LOCATED IN AUX1.MAC (SECTION L)

POLISH = EVALUATES POLISH EXPRESSIONS, LOCATED IN CALC.MAC (SECTION D)

QCHD = EXECUTES REPEAT QUESTION (Q1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION F)

RCHD = EXECUTES REPEAT (R1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION D)

RETSUB = RETURNS FROM A SUBROUTINE, LOCATED IN ACTION.MAC (SECTION J)

SCDH = EXECUTES SLIDE (S1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION G)

UCCHD = EXECUTES CALC (G1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION C)

UFCHD = EXECUTES FEEDBACK (F1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION A)

UPCHD = EXECUTES PAGE (P1) STATEMENTS, LOCATED IN ACTION.MAC (SECTION L)

USCHD = 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 = GETSEQ 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

HINTNT = NO. OF HINTS REQUESTED AND NO. OF HINT-TAGS ENCOUNTERED IN THE ACTION SECTION (2 WORDS)

IBUF = DISK INPUT BUFFER HEADER (3 WORDS)

IDXSAV = DATA INDEX IS SAVED HERE WHEN AQ-F IS USED BY ANOTHER SUBPROGRAM

IDX = INDEX TO THE 1ST WORD CONTAINING STUDENT PERFORMANCE DATA

INAM = DISK LOOKUP ARGUMENTS (4 WORDS)

LANS = LESSON ANSWER BUFFER (16 WORDS)

LESHRD = 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
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

TRMBLK = TRMOP, ARGUMENTS (3 WORDS)

WNUM = NO. OF WRONG ANSWERS

DESCRIPTION OF FLAGS STORED IN AC=8

BIT 0 TEST LESSON
BIT 1 INFORMATION LESSON
BIT 2 NOT USED
BIT 3 CORRECT ANSWER
BIT 4 WRONG ANSWER
BIT 5 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
<table>
<thead>
<tr>
<th>BIT</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>REVIEW SLIDE</td>
</tr>
<tr>
<td>27</td>
<td>NOT USED</td>
</tr>
<tr>
<td>28</td>
<td>REVIEW IN PROGRESS</td>
</tr>
<tr>
<td>29</td>
<td>REVIEW REQUESTED</td>
</tr>
<tr>
<td>30</td>
<td>NOT USED</td>
</tr>
<tr>
<td>31</td>
<td>ANSWER FILE OPEN</td>
</tr>
<tr>
<td>32</td>
<td>NOT USED</td>
</tr>
<tr>
<td>33</td>
<td>NOT USED</td>
</tr>
<tr>
<td>34</td>
<td>NOT USED</td>
</tr>
<tr>
<td>35</td>
<td>ZERO TAG</td>
</tr>
</tbody>
</table>

**DESCRIPTION OF OFLAGS STORED IN LOW SEGMENT**

<table>
<thead>
<tr>
<th>BIT</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-19</td>
<td>NOT USED</td>
</tr>
<tr>
<td>20</td>
<td>RECORD ON</td>
</tr>
<tr>
<td>21</td>
<td>REVIEW ON</td>
</tr>
<tr>
<td>22</td>
<td>TEMPORARY REVIEW ON</td>
</tr>
<tr>
<td>23</td>
<td>TEMPORARY REVIEW</td>
</tr>
<tr>
<td>24</td>
<td>STORE ON</td>
</tr>
<tr>
<td>25</td>
<td>TEMPORARY STORE ON</td>
</tr>
<tr>
<td>26</td>
<td>TEMPORARY STORE</td>
</tr>
<tr>
<td>27</td>
<td>BOTH ON</td>
</tr>
<tr>
<td>28</td>
<td>TEMPORARY BOTH ON</td>
</tr>
<tr>
<td>29</td>
<td>TEMPORARY BOTH</td>
</tr>
<tr>
<td>30</td>
<td>HINT ON</td>
</tr>
<tr>
<td>31</td>
<td>TEMPORARY HINT ON</td>
</tr>
<tr>
<td>32</td>
<td>TEMPORARY HINT</td>
</tr>
<tr>
<td>33</td>
<td>CALC ON</td>
</tr>
<tr>
<td>34</td>
<td>TEMPORARY CALC ON</td>
</tr>
<tr>
<td>35</td>
<td>TEMPORARY CALC ON</td>
</tr>
</tbody>
</table>

**DESCRIPTION OF OFLAGS STORED IN AO-16**

<table>
<thead>
<tr>
<th>BIT</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0-FRAME</td>
</tr>
<tr>
<td>25</td>
<td>NEGATIVE NUMBER</td>
</tr>
</tbody>
</table>

**HISEG**

<table>
<thead>
<tr>
<th>ENTRY</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERP</td>
<td>FSTLIN; ANSH, GETACT, OFRAM1, UNCA1, EOL</td>
</tr>
<tr>
<td>INTERP</td>
<td>LOGOUT, SKPTXT, SDLRTN, FAS</td>
</tr>
<tr>
<td>INTERP</td>
<td>STDATE, GETBUF, EXPDOR, TMBLK, CMD</td>
</tr>
<tr>
<td>INTERP</td>
<td>CONNUM, SVPOS, RGANS, FINT, UPCHD, PCHD</td>
</tr>
<tr>
<td>INTERP</td>
<td>TAGREC, TAGCNT, TAGREP, TAPNT, ADDR, SYMIX</td>
</tr>
<tr>
<td>INTERP</td>
<td>RANNUM, SNANS, INFPL, POLISH, POLLIST, IDXSAY, TOL, IDIX</td>
</tr>
<tr>
<td>INTERP</td>
<td>HNTCFT, SUBDOR, REVIDX, REVNUM, SAVPOS, RSVNUM</td>
</tr>
<tr>
<td>INTERP</td>
<td>APYT, STORD, LESWRD, SANSR, PCTNT, OFRAM1, ARGMT</td>
</tr>
<tr>
<td>INTERP</td>
<td>TIME, INAM, ONAM, OUTCHR, CALC, DELIM, COUNT, DEVE</td>
</tr>
</tbody>
</table>
INTERP MACRO 47(204)-7 00106 2-AUG-74 PAGE 1-8

EXTERN FRMTOT,RNUM,WNUM,WNUM,THNUM,DATA,RTUNTH,ANSTM
EXTERN I0D,10T,SDLNUM,TAG,STORE,BLOCK,IBUF
EXTERN SANS,LANS,FRNUM,FRNUM,FRNTYP,PC2,PPN
EXTERN ACMD,BCMD,BCMD2,CMD2,CMD2,FCHD,FCMD,CCMD,GCMD,RCMD
EXTERN RSTSUB,SCMD,UCMD,UPCMD,OFLAGS,LCMD,USCMD,BYE,GETSG

000000 000000\ Flags$=8
000001 000002 000003 000004 000005 \#1
000006 000007 000008 000009 00000A \#2
00000B 00000C 00000D 00000E 00000F \#3
000010 000011 000012 000013 000014 \#4
000015 000016 000017 000018 000019 \#5

00001A 00001B 00001C 00001D 00001E \#6
00001F 000020 000021 000022 000023 \#7
000024 000025 000026 000027 000028 \#8
000029 00002A 00002B 00002C 00002D \#9
00002E 00002F 000030 000031 000032 \#0

KI1B=1 IF FOR KI1B CPU, LET KI1B=0

CRLF$ ASCIZ "/

IDISK INPUT ROUTINE WHICH IS TRANSFERRED TO AC-11 THRU AC-15
RTN 15 15,GETBUF IF BUFFER IS EMPTY, GO FILL IT (AUX1,MAC)
IDOB A,14 GET CHARACTER FROM BUFFER
MOVEM 15,SAVPOS=2  JSAVE BUFFER BYTE COUNT
FSTL1 AOS INDEX  JINCREMENT DATA INDEX
JSP IF XPOSOR  JEXPAND CORE IF NECESSARY
JSP Q:11D  JINPUT FRAME NO.
MOVEM A,FRMTYP  JSAVE FRAME TYPE IN FRMTYP
FRM 33A,POINT  JSTORE FRAME TYPE IN STUDENT PERFORMANCE DATA
CAHN B,SAVPOS=3  JBE HISTORY WHERE A REVIEW STARTED
FRTS 2,22B  JYES, CLEAR REVIEW IN PROGRESS FLAG
MOVEM B,FRNMUN  JIS SAVE FRAME NO. IN FRMNUM
FRTS 2,22B  JIS REVIEW IN PROGRESS?
ADDI B=+1200  JYES, ADD $200 ONTO FRAME NO.
FRN 33A,POINT  JIS STORE FRAME NO. WITH STUDENT PERFORMANCE DATA
TLN 40B,22B  JIS TEST MODE IN EFFECT?
JRT FSTL3  JNO, GO TO FSTL3
JRCALL 5,CRLDF
JRCALL 8,CASC12
JFM, FF
MOVEM A,FRMNUM
JSP Q:10T  JTYPE FRAME NUMBER
JRCALL 5,CRLDF
FRM 33A,POINT
FSTL31 AOS FRMTOT  JINCREMENT FRAME TOTAL
HSTIME B, JGET TIME OF DAY
DIVI B=+1200  JCONV TO SECONDS
HRHM B,DATA (INDEX)  JADD TIME TO STUDENT PERFORMANCE DATA
MOVEM A,FRMTYP  JIS FRAME TYPE IN AC=4
FRTS 2,22B  JIS THIS A Q-FRAME?
CAIN A, "CM"  JYES, GO TO OFRAME (SECTION C)
CAIN A, "CH"  JIS THIS A C-FRAME?
CAIN A, "CFRAME"  JYES, GO TO C-FRAME (CALC,MAC)
CAIN A, "RFRAME"  JYES, GO TO OFRAME (SECTION E)
CAIN A, "RFRAME"  JYES, GO TO R-FRAME (SECTION D)
CAIN A, "RFRAME"  JYES, GO TO BFFRAME (SECTION D)

SECTION C, EVALUATE THE 7 BIT CODE AT THE BEGINNING OF A Q FRAME
STATEMENT AND BRANCH TO THE ROUTINE WHICH EXECUTES THE REST OF
THE STATEMENT.

MOVEM A,OFLGS  JSAVE THE FLAGS
SETZ TAGCNT  JSET TAG COUNT TO ZERO
MOVEM B,POINT  JSAVE THE POINTERS
MOVEM B,POINT  JPUT TAG POINTER IN TAGPLNT
SETZ TAGREG  JSET TAG REGISTER
SETZ TAGREG=1
The routine below stores correct answer for A1 action:

- `!!TRANSI SET A1, ICLEAR AC-A FOR ROUTINE AT END`:
- `!!TFL ASK2.20 : HAS ANSWER ALREADY BEEN STORED?`
- `!!JST FNDAC IF YES, GO TO FNDAC (SECTION 2)`
- `!!MOVE B,PONT15 GET POINTER TO RTANS IN AC-B`:
- `!!MOVE B,CASC1 /TIMEUP`
- `!!MOVE C,RTANS MOVE RETURN CHARACTER TO RTANS`:
- `!!JSP 0.1101 INPUT CHARACTER`:
- `!!CASH 0.00094 END OF LESSON?`
- `!!JST EOL IF YES, GO TO EOL (SECTION MM)`
- `!!CASH 0.00093 BEGINNING OF NEXT FRAME?`
- `!!JST FS1LIN IF YES, GO TO FS1LIN (SECTION B)`
- `!!ACTION STATEMENT?`:
- `!!JST FNDAC IF YES, GO TO FNDAC (SECTION 2)`
- `!!CA1E 11`:
- `!!CA1E 13`:
- `!!JST SKPSET IF <11> OR <13>, GO TO SKPSET`:
- `!!CA1E 15`:
- `!!CASH 0.00094`:
- `!!JST SKPSET IF CHARACTER IS <20> OR <14>, GO TO SKPSET`:
- `!!CA1E 20`:
- `!!CA1E 23`:
- `!!CA1E 28`:
- `!!JST SKPSET IF <20> OR <23>, GO TO SKPSET`:
- `!!CA1E A""`:
- `!!CA1E A""`:
- `!!SET NUMERIC LESSON ANSWER FLAG`:
- `!!INPUT RIGHT-WRONG-NEUTRAL INDICATOR`:
- `!!CA1E A"" IS THIS A CORRECT ANSWER?`:
- `!!JST SKPSET JOB, SKIP IT`
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1114</td>
<td>001104</td>
<td>2BA 00 00 000104*</td>
</tr>
<tr>
<td>1115</td>
<td>001105</td>
<td>2BA 00 00 000101*</td>
</tr>
<tr>
<td>1116</td>
<td>001106</td>
<td>2BA 00 00 000102*</td>
</tr>
<tr>
<td>1117</td>
<td>001107</td>
<td>2BA 00 00 000103*</td>
</tr>
<tr>
<td>1118</td>
<td>001108</td>
<td>2BA 00 00 000104*</td>
</tr>
<tr>
<td>1119</td>
<td>001109</td>
<td>2BA 00 00 000105*</td>
</tr>
<tr>
<td>1120</td>
<td>001110</td>
<td>2BA 00 00 000106*</td>
</tr>
<tr>
<td>1121</td>
<td>001111</td>
<td>2BA 00 00 000107*</td>
</tr>
<tr>
<td>1122</td>
<td>001112</td>
<td>2BA 00 00 000108*</td>
</tr>
<tr>
<td>1123</td>
<td>001113</td>
<td>2BA 00 00 000109*</td>
</tr>
<tr>
<td>1124</td>
<td>001114</td>
<td>2BA 00 00 000100*</td>
</tr>
<tr>
<td>1125</td>
<td>001115</td>
<td>2BA 00 00 000101*</td>
</tr>
<tr>
<td>1126</td>
<td></td>
<td>ESI: MOVE A, OFLAGS, IGET, FLGS IN AC-E</td>
</tr>
<tr>
<td>1127</td>
<td></td>
<td>TRUE A: 1 IIS TEMP. CALC SET?</td>
</tr>
<tr>
<td>1128</td>
<td></td>
<td>JRST: +4 YES, SKIP</td>
</tr>
<tr>
<td>1129</td>
<td></td>
<td>JRT: +4 YES, SKIP</td>
</tr>
<tr>
<td>1130</td>
<td></td>
<td>JRST: NOCALC, NO ERROR. MESSAGE</td>
</tr>
<tr>
<td>1131</td>
<td></td>
<td>TRNN A: 2 IIS TEMP. CALC ON SET?</td>
</tr>
<tr>
<td>1132</td>
<td></td>
<td>JRT: NOCALC, NO ERROR. MESSAGE</td>
</tr>
<tr>
<td>1133</td>
<td></td>
<td>JRT: NOCALC, NO ERROR. MESSAGE</td>
</tr>
<tr>
<td>1134</td>
<td></td>
<td>JRT: NODATA (INDEX) SET CALC BIT IN STUD., PERI. DATA</td>
</tr>
<tr>
<td>1135</td>
<td></td>
<td>TCALL 3, CASCIE /</td>
</tr>
<tr>
<td>1136</td>
<td></td>
<td>ANSWER</td>
</tr>
<tr>
<td>1137</td>
<td></td>
<td>MOVE E, ANSTIM=2</td>
</tr>
<tr>
<td>1138</td>
<td></td>
<td>INPUT WAIT TIME BACK IN AC-E</td>
</tr>
<tr>
<td>1139</td>
<td></td>
<td>JUMPE E, ANSIN IF INFINITE WAIT TIME, GO TO ANSIN (SECT. 0)</td>
</tr>
<tr>
<td>1140</td>
<td></td>
<td>JRT: WAIT, ELSE, CONTINUE TO WAIT FOR THE ANSWER, (SECT. 0)</td>
</tr>
<tr>
<td>1141</td>
<td></td>
<td>NOCALC TCALL 3, CASCIE /</td>
</tr>
<tr>
<td>1142</td>
<td></td>
<td>NO CALCATION NOT AVAILABLE, PLEASE TYPE THE ANSWER.</td>
</tr>
<tr>
<td>1143</td>
<td></td>
<td>MOVE E, ANSTIM=2 GET THE WAIT TIME</td>
</tr>
<tr>
<td>1144</td>
<td></td>
<td>JUMPE E, ANSIN IF INFINITE WAIT TIME, GO TO ANSIN (SECT. 0)</td>
</tr>
<tr>
<td>1145</td>
<td></td>
<td>JRT: WAIT, ELSE, CONTINUE TO WAIT FOR THE ANSWER, (SECT. 0)</td>
</tr>
<tr>
<td>1146</td>
<td>001127</td>
<td>2BA 00 00 000100*</td>
</tr>
<tr>
<td>1147</td>
<td>001128</td>
<td>2BA 00 00 000101*</td>
</tr>
<tr>
<td>1148</td>
<td>001129</td>
<td>2BA 00 00 000102*</td>
</tr>
<tr>
<td>1149</td>
<td>001130</td>
<td>2BA 00 00 000103*</td>
</tr>
<tr>
<td>1150</td>
<td>001131</td>
<td>2BA 00 00 000104*</td>
</tr>
<tr>
<td>1151</td>
<td>001132</td>
<td>2BA 00 00 000105*</td>
</tr>
<tr>
<td>1152</td>
<td>001133</td>
<td>2BA 00 00 000106*</td>
</tr>
<tr>
<td>1153</td>
<td>001134</td>
<td>2BA 00 00 000107*</td>
</tr>
<tr>
<td>1154</td>
<td>001135</td>
<td>2BA 00 00 000108*</td>
</tr>
<tr>
<td>1155</td>
<td>001136</td>
<td>2BA 00 00 000109*</td>
</tr>
<tr>
<td>1156</td>
<td>001137</td>
<td>2BA 00 00 000100*</td>
</tr>
<tr>
<td>1157</td>
<td>001138</td>
<td>2BA 00 00 000101*</td>
</tr>
<tr>
<td>1158</td>
<td>001139</td>
<td>2BA 00 00 000102*</td>
</tr>
<tr>
<td>1159</td>
<td>001140</td>
<td>2BA 00 00 000103*</td>
</tr>
<tr>
<td>1160</td>
<td>001141</td>
<td>2BA 00 00 000104*</td>
</tr>
<tr>
<td>1161</td>
<td>001142</td>
<td>2BA 00 00 000105*</td>
</tr>
<tr>
<td>1162</td>
<td>001143</td>
<td>2BA 00 00 000106*</td>
</tr>
<tr>
<td>1163</td>
<td>001144</td>
<td>2BA 00 00 000107*</td>
</tr>
<tr>
<td>1164</td>
<td>001145</td>
<td>2BA 00 00 000108*</td>
</tr>
<tr>
<td>1165</td>
<td>001146</td>
<td>2BA 00 00 000109*</td>
</tr>
<tr>
<td>1166</td>
<td>001147</td>
<td>2BA 00 00 000100*</td>
</tr>
<tr>
<td>1167</td>
<td>001148</td>
<td>2BA 00 00 000101*</td>
</tr>
<tr>
<td>1168</td>
<td>001149</td>
<td>2BA 00 00 000102*</td>
</tr>
<tr>
<td>1169</td>
<td>001150</td>
<td>2BA 00 00 000103*</td>
</tr>
<tr>
<td>1170</td>
<td>001151</td>
<td>2BA 00 00 000104*</td>
</tr>
<tr>
<td>1171</td>
<td>001152</td>
<td>2BA 00 00 000105*</td>
</tr>
<tr>
<td>1172</td>
<td>001153</td>
<td>2BA 00 00 000106*</td>
</tr>
<tr>
<td>1173</td>
<td>001154</td>
<td>2BA 00 00 000107*</td>
</tr>
<tr>
<td>1174</td>
<td></td>
<td>CLEAR NUMERIC ANSWER FLAG,��</td>
</tr>
</tbody>
</table>
1220 001235 200 02 0 0 000001
1221 001236 202 06 0 0 0000004
1222 001237 202 03 0 0 000007
1223 001240 626 16 0 0 002000
1224 001241 134 01 0 0 000002
1225 001242 382 01 0 0 000000
1226 001243 254 00 0 0 001246
1227 001244 200 07 0 0 002000
1228 001245 254 00 0 0 001243
1229 001246 382 01 0 0 002000
1230 001247 254 00 0 0 001252
1231 001250 626 16 0 0 002000
1232 001251 200 07 0 0 002003
1233 001252 265 17 0 0 0000006
1234 001253 255 00 0 0 002000
1235 001254 200 06 0 0 001256
1236 001255 255 01 0 0 001257
1237 001256 254 00 0 0 001266
1238 001257 265 03 0 0 001260
1239 001258 255 17 0 0 001261
1240 001259 603 03 0 0 001100
1241 001262 254 00 0 0 001265
1242 001263 200 02 0 0 002367
1243 001264 254 00 0 0 001266
1244 001265 200 02 0 0 003708
1245 001266 626 22 16 0 0 002000
1246 001267 213 00 0 0 000002
1247 001268 200 02 0 0 0020008
1248 001269 661 00 0 0 002000
1249 001270 660 00 0 0 002001
1250 001271 254 00 0 0 001171
1251 001274 254 00 0 0 001301
1252
1253
1254
1255
1256
1257 001275 603 00 0 0 000040
1258 001276 254 00 0 0 001301
1259 001277 603 00 0 0 000001
1260 001280 254 00 0 0 001244
1261 001281 661 16 0 0 000000
1262 001282 254 17 0 0 000001
1263 001283 254 00 0 0 001311
1264
1265
1266
1267
1268
1269
1270
1271
1272

SECTION K. INTERPRET CALC STATEMENTS AFTER "B" TAGS.
1273
1274
1275
1276
1277
1278

SECTION L. INTERPRET EXTRA, ORDER, AND PHONIC STATEMENTS
1279
1280
1281
1282

INTERPRET EXTRA, ORDER, AND PHONIC STATEMENTS
SECTION M. INTERPRET MATCH STATEMENTS AND PLACE THE MATCH NUMBER IN COUNTER.

- **MTSET** JSP Q,11 INPUT SINGLE DIGIT
- **SUB1** A,40 SUBTRACT TO GET VALUE OF DIGIT
- **MOVEH A,COUNT** SAVE NO. OF WORDS REQUIRED FOR A MATCH
- **JSP Q,11** INPUT RETURN CHARACTER
- **JST EVAL2** GOTO TO EVAL2 (SECTION I)

SECTION N. INTERPRET DELIM STATEMENTS.

- **DELS** JSP Q,11 INPUT CHARACTER
- **CAIN A,15** END OF STATEMENT?
- **JST SDDEL** YES, GO SET UP LIST OF STANDARD DELIMITERS
- **MOVEI 9,1** SET INDEX TO 1
- **MOVEH A,DELIM** MOVE 1ST DELIMITER TO LIST OF DELIMITERS
- **JSP Q,11** INPUT NEXT DELIMITER
- **CAIN A,15** END OF STATEMENT?
- **JST ENDEL** YES, GO TO ENDEL
- **MOVEH A,DELIM+1** INPUT DELIMITER IN LIST OF DELIMITERS
- **AQA A, -1** ID GET NEXT DELIMITER
- **ENDDEL** SET DELIM (B) ZERO INDICATES END OF DELIMITER LIST
- **JST EVAL2** GOTO TO EVAL2 (SECTION I)
- **STDDLM** JSP Q,STDDLM SET UP LIST OF STANDARD DELIMITERS (SECTION I)

SECTION O. SUBROUTINE WHICH SETS UP LIST OF STANDARD WORD DELIMITERS.

- **STDDLM** MOVEI A,40 MAKE A SPACE A DELIMITER
- **MOVEH A, ."** MAKE A PERIOD A DELIMITER
- **MOVEH A, ."** MAKE A COMMA A DELIMITER
- **MOVEH A, :) MAKE A SEMICOLON A DELIMITER
- **MOVEH A, :) MAKE A COLON A DELIMITER
- **MOVEH A, :) MAKE AN EXCLAMATION POINT A DELIMITER
- **MOVEH A, ?** MAKE A QUESTION MARK A DELIMITER
- **MOVEH A, :) MAKE A QUESTION MARK A DELIMITER
- **JST (Q) RETURN
SECTION P. TEST ANSWER EVALUATION MODE FLAGS AND BRANCH TO THE PROPER ROUTINE TO COMPARE LESSON ANSWERS TO THE STUDENT'S ANSWER.

SECTION Q. COMPARE LESSON ANSWER AND STUDENT ANSWER WITH EXTRA OFF AND ORDER ON.

SECTION R. COMPARE LESSON AND STUDENT ANSWERS WITH EXTRA OFF AND ORDER OFF.
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
INTERP MACRO 47(204)-7 20126 2-AUG-74 PAGE 1-32

1607 02054 306 03 00 00138 CAIN A, "X"
1608 02059 254 00 00 00211 JSTR XXX
1609 02059 256 01 00 00124 CAIN A, "T"
1700 02057 201 03 00 00113 MOVEI A, "K"
1701 02057 156 04 00 0003 P31 IDPB A,C
1702 02057 205 03 00 00036 CAIN A, "M"
1703 02057 254 00 00 002130 JSTR ENDOHK
1704 02057 300 01 00 00121 CAIL A, "A"
1705 02057 303 02 00 00132 CAIL A, "P"
1706 02057 256 00 00 00176 JSTR P1
1707 02057 256 00 00 00215 JSTR P2
1708 02057 205 02 00 00001 OMIT MOVE G,A
1709 02057 254 00 00 002150 JSTR P2
1710 02057 134 01 00 00024 JSGS ILOA A,B
1711 02057 306 01 00 00015 CAIN A, "N"
1712 02057 256 00 00 002130 JSTR ENDOHK
1713 02057 301 01 00 00121 CAIL A, "A"
1714 02057 303 01 00 00132 CAIL A, "M"
1715 02057 254 00 00 00176 JSTR P1S
1716 02057 200 04 00 00124 MOV D,A
1717 02057 201 01 00 00113 IDPB A,C
1718 02057 136 00 00023 JSTR P2
1719 02057 200 01 00 00244 MOVER A,D
1720 02057 254 00 00 00216 JSTR P2+1
1721 02057 201 01 00 00186 MOVER A, "F"
1722 02057 254 00 00 00216 JSTR P3
1723 02057 306 07 00 00125 NNNI CAIN G, "F"
1724 02057 254 00 00 00216 JSTR OMIT
1725 02057 254 00 00 00216 JSTR P3
1726 02057 211 01 00 00113 XXI MOVER A, "K"
1727 02057 136 01 00 00205 IDPB A,C
1728 02057 254 00 00 00216 JSTR P3
1729 02057 306 07 00 00125 KAYI CAIN G, "I"
1730 02057 254 00 00 00216 JSTR OMIT
1731 02057 254 00 00 00216 JSTR P3
1732 02057 203 07 00 00011 MOVER A, "G"
1733 02057 254 00 00 00216 JSTR P3
1734 02057 302 07 00 00113 GEET CAIN G, "I"
1735 02057 302 07 00 00113 CAIN G, "N"
1736 02057 254 00 00 00216 JSTR OMIT
1737 02057 254 00 00 00216 JSTR OMIT
1738 02057 203 07 00 00011 MOVER A, "P"
1739 02057 254 00 00 00216 JSTR P3
1740 02057 139 01 00 00003 IDPB A,C
1741 02057 254 00 00 00216 JSTR P3
1742 02057 139 01 00 00003 ENDOHK
1743 02057 254 00 00 00216 IDPB A,C
1744 02057 254 00 00 00001 TLO FLAGS,1
1745 02057 294 00 00 00001 JSTR (Q)
1746 02057 017 00 00000 RETURN

SECTION X. THE SUBROUTINES BELOW ARE CALLED BY THE ANSWER EVALUATION
SUBROUTINES.
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>MOV D:B</td>
</tr>
<tr>
<td>104</td>
<td>SAVE POINTER IN AC:B</td>
</tr>
<tr>
<td>105</td>
<td>MOV A:D</td>
</tr>
<tr>
<td>106</td>
<td>GET NEXT CHARACTER IN AC:A</td>
</tr>
<tr>
<td>107</td>
<td>MOV A:A</td>
</tr>
<tr>
<td>108</td>
<td>CALE A, A M</td>
</tr>
<tr>
<td>109</td>
<td>JUMP IF CHARACTER IS PLUS OR MINUS</td>
</tr>
<tr>
<td>110</td>
<td>MOV B:D</td>
</tr>
<tr>
<td>111</td>
<td>PUT POINTER IN AC:B</td>
</tr>
<tr>
<td>112</td>
<td>JUMP IF NOT A NUMERAL, GO TO NUMED</td>
</tr>
<tr>
<td>113</td>
<td>MOV E:D</td>
</tr>
<tr>
<td>114</td>
<td>PUT E:0 ADDRESS</td>
</tr>
<tr>
<td>115</td>
<td>JSP Q, BACKUP</td>
</tr>
<tr>
<td>116</td>
<td>BACKUP ONE CHARACTER</td>
</tr>
<tr>
<td>117</td>
<td>JSP E:RESULT</td>
</tr>
<tr>
<td>118</td>
<td>BACKUP ONE CHARACTER</td>
</tr>
<tr>
<td>119</td>
<td>JSP Q, BACKUP</td>
</tr>
<tr>
<td>120</td>
<td>CLEAR THE LESSON WORD HOLDER</td>
</tr>
<tr>
<td>121</td>
<td>CLEAR 1ST WORD OF LESSON WORD HOLDER</td>
</tr>
<tr>
<td>122</td>
<td>CLEAR 2ND WORD</td>
</tr>
<tr>
<td>123</td>
<td>CLEAR 3RD WORD</td>
</tr>
<tr>
<td>124</td>
<td>CLEAR LAST WORD</td>
</tr>
<tr>
<td>125</td>
<td>JSP Q, RETURN</td>
</tr>
<tr>
<td>126</td>
<td>CLEAR THE STUDENT WORD HOLDER</td>
</tr>
<tr>
<td>127</td>
<td>CLEAR 1ST WORD OF STUDENT WORD HOLDER</td>
</tr>
<tr>
<td>128</td>
<td>CLEAR 2ND WORD</td>
</tr>
<tr>
<td>129</td>
<td>CLEAR 3RD WORD</td>
</tr>
<tr>
<td>130</td>
<td>CLEAR LAST WORD</td>
</tr>
<tr>
<td>131</td>
<td>JSP Q, RETURN</td>
</tr>
<tr>
<td>132</td>
<td>I ADVANCE LESSON ANSWER BYTE POINTER TO NEXT WORD</td>
</tr>
<tr>
<td>133</td>
<td>MOV E:0, MOV E:1</td>
</tr>
<tr>
<td>134</td>
<td>SET AC:D TO 1 (INDICATES LESSON ANSWER)</td>
</tr>
<tr>
<td>135</td>
<td>TLNE FLAGS, 4</td>
</tr>
<tr>
<td>136</td>
<td>END OF LESSON ANSWER</td>
</tr>
<tr>
<td>137</td>
<td>JSP Q, YES, RETURN</td>
</tr>
<tr>
<td>138</td>
<td>JSP ST I Go TO ST</td>
</tr>
<tr>
<td>139</td>
<td>I ADVANCE STUDENT ANSWER BYTE POINTER TO NEXT WORD</td>
</tr>
<tr>
<td>140</td>
<td>MOV E:0, MOV E:1</td>
</tr>
<tr>
<td>141</td>
<td>SET AC:D TO ZERO (INDICATES STUDENT ANSWER)</td>
</tr>
<tr>
<td>142</td>
<td>TLNE FLAGS, 2</td>
</tr>
<tr>
<td>143</td>
<td>END OF STUDENT ANSWER</td>
</tr>
<tr>
<td>144</td>
<td>JSP Q, YES, RETURN</td>
</tr>
<tr>
<td>145</td>
<td>I GDP A:B</td>
</tr>
<tr>
<td>146</td>
<td>PUT ANSWER CHARACTER IN AC:A</td>
</tr>
<tr>
<td>147</td>
<td>JSP Q, YES, RETURN</td>
</tr>
<tr>
<td>148</td>
<td>YES, GO TO ENDANS</td>
</tr>
<tr>
<td>149</td>
<td>SET E:G, SET INDEX TO ZERO</td>
</tr>
<tr>
<td>150</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
<tr>
<td>151</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
<tr>
<td>152</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
<tr>
<td>153</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
<tr>
<td>154</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
<tr>
<td>155</td>
<td>JSP Q, YES, GO TO BACKUP</td>
</tr>
</tbody>
</table>
ISECTION CC. STORE STUDENT PERFORMANCE DATA.

ISECTION DD. INTERPRET B-FRAMES.
INTERP MACRO 47 (204) - 2 0030 6 2-AUG-74 PAGE 1-50
INTERP MAC 30-JUL-74

2651 003571 205 01 0 0 0 0447000
2652 003571 202 01 0 0 0 000002
2653 003571 402 00 0 0 000063
2654 003572 055 01 0 0 000255
2655 003573 255 00 0 0 000000
2656

2657 003574 637 00 0 0 0 000001
2659 003575 254 00 0 0 0 000031
2661 003576 254 00 0 0 0 000000

NO ERRORS DETECTED

PROGRAM BREAK IS 004031

4K CORE USED
<table>
<thead>
<tr>
<th></th>
<th>1499</th>
<th>1506</th>
<th>1508</th>
<th>1512</th>
<th>1518</th>
<th>1525</th>
<th>1532</th>
<th>1534</th>
<th>1538</th>
<th>1545</th>
<th>1548</th>
<th>1550</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1529</td>
<td>1535</td>
<td>1537</td>
<td>1544</td>
<td>1571</td>
<td>1573</td>
<td>1579</td>
<td>1577</td>
<td>1590</td>
<td>1593</td>
<td>1599</td>
<td>1603</td>
</tr>
<tr>
<td>FNDAC1</td>
<td>1592</td>
<td>1594</td>
<td>1596</td>
<td>1606</td>
<td>1608</td>
<td>1610</td>
<td>1612</td>
<td>1614</td>
<td>1621</td>
<td>1639</td>
<td>1742</td>
<td>1753</td>
</tr>
<tr>
<td>FNDAC2</td>
<td>1647</td>
<td>1673</td>
<td>1679</td>
<td>1694</td>
<td>1697</td>
<td>1699</td>
<td>1708</td>
<td>1714</td>
<td>1721</td>
<td>1723</td>
<td>1739</td>
<td>1740</td>
</tr>
<tr>
<td>FNDACR</td>
<td>2474</td>
<td>2476</td>
<td>2607</td>
<td>2658</td>
<td>2676</td>
<td>2709</td>
<td>2721</td>
<td>2731</td>
<td>2733</td>
<td>2735</td>
<td>2736</td>
<td>2737</td>
</tr>
<tr>
<td>FNDAC1</td>
<td>1199</td>
<td>1945#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FNDAC2</td>
<td>1936#</td>
<td>1942</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FNDACR</td>
<td>981</td>
<td>991</td>
<td>1013</td>
<td>1212</td>
<td>1934#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRI</td>
<td>1496</td>
<td>1514#</td>
<td>1533</td>
<td>1556</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRI1</td>
<td>1522#</td>
<td>1549</td>
<td>1563</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRI2</td>
<td>1592#</td>
<td>1609</td>
<td>1613</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRIE</td>
<td>1391</td>
<td>1399#</td>
<td>1471#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRIE1</td>
<td>1396</td>
<td>1436</td>
<td>1471#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRIE2</td>
<td>1397</td>
<td>1407</td>
<td>1480#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRESUB</td>
<td>1428</td>
<td>1460</td>
<td>1562#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRHNUM</td>
<td>427#</td>
<td>522</td>
<td>685</td>
<td>2561</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRMTYP</td>
<td>427#</td>
<td>588</td>
<td>611</td>
<td>2424</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRSUB</td>
<td>1411</td>
<td>1471</td>
<td>1497#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRSUB1</td>
<td>1419</td>
<td>1488#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSY</td>
<td>1461</td>
<td>1274#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSTLP</td>
<td>594</td>
<td>598#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSTL2</td>
<td>589#</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>FSTL3</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>
</tr>
<tr>
<td>FSTLIN</td>
<td>419</td>
<td>568</td>
<td>576#</td>
<td>595</td>
<td>655</td>
<td>989</td>
<td>1193</td>
<td>1990</td>
<td>2018</td>
<td>2097</td>
<td>2187</td>
<td>2122</td>
</tr>
<tr>
<td>FWOS</td>
<td>79#</td>
<td>823</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>43#</td>
<td>611</td>
<td>693</td>
<td>788</td>
<td>1222</td>
<td>1222</td>
<td>1227</td>
<td>1232</td>
<td>1640</td>
<td>1679</td>
<td>1687</td>
<td>1788</td>
</tr>
<tr>
<td>1731</td>
<td>1734</td>
<td>1736</td>
<td>1738</td>
<td>1852</td>
<td>1855</td>
<td>1857</td>
<td>1859</td>
<td>1996</td>
<td>1998</td>
<td>1932</td>
<td>1918</td>
<td></td>
</tr>
<tr>
<td>2133</td>
<td>2181</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>GCOM</td>
<td>428#</td>
<td>2056#</td>
<td>2886</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEE</td>
<td>1636</td>
<td>1733#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETACI</td>
<td>2039</td>
<td>2056#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETACT</td>
<td>419</td>
<td>2059#</td>
<td>2109</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETBUF</td>
<td>418#</td>
<td>476</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETS</td>
<td>242#</td>
<td>2442</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRTRAN</td>
<td>919</td>
<td>1106</td>
<td>1421</td>
<td>1452</td>
<td>1473</td>
<td>1482</td>
<td>1509</td>
<td>1574</td>
<td>1578</td>
<td>1581</td>
<td>1922</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>43#</td>
<td>2160</td>
<td>2166</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HINT1</td>
<td>939</td>
<td>1069#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HINTC</td>
<td>422#</td>
<td>639</td>
<td>1097</td>
<td>1211</td>
<td>1599</td>
<td>1981</td>
<td>2847</td>
<td>2849</td>
<td>2849</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HINTSET</td>
<td>653</td>
<td>676#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBUF</td>
<td>421#</td>
<td>2125</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDASAV</td>
<td>421#</td>
<td>1235</td>
<td>1911</td>
<td>1913</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IID</td>
<td>426#</td>
<td>587</td>
<td>2266</td>
<td>2423</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDDX</td>
<td>421#</td>
<td>537</td>
<td>2529</td>
<td>2534</td>
<td>2568</td>
<td>2572</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INAM</td>
<td>424#</td>
<td>548</td>
<td>718</td>
<td>2222</td>
<td>2460</td>
<td>2484</td>
<td>2488</td>
<td>2496</td>
<td>2492</td>
<td>2494</td>
<td>2497</td>
<td>2501</td>
</tr>
<tr>
<td>INDEX</td>
<td>2613</td>
<td>2617</td>
<td>2619</td>
<td>2641</td>
<td>2642</td>
<td>2643</td>
<td>2650</td>
<td>2652</td>
<td>2654</td>
<td>2659</td>
<td>2655</td>
<td>2664</td>
</tr>
<tr>
<td>INFIP</td>
<td>421#</td>
<td>1233</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QCMOD</td>
<td>426#</td>
<td>2269#</td>
<td>413#</td>
<td>639#</td>
<td>661#</td>
<td>668#</td>
<td>669#</td>
<td>708#</td>
<td>717#</td>
<td>727#</td>
<td>738#</td>
<td>748#</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>
</tr>
<tr>
<td>QCMND</td>
<td>629#</td>
<td>176#</td>
<td>1769#</td>
<td>1795#</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
<td>478#</td>
<td>509#</td>
<td>524#</td>
<td>527#</td>
<td>531#</td>
</tr>
<tr>
<td>PNP</td>
<td>427#</td>
<td>2159#</td>
<td>2493#</td>
<td>2539#</td>
<td>2621#</td>
<td>2626#</td>
<td>176#</td>
<td>1769#</td>
<td>1795#</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
</tr>
<tr>
<td>SCHOFF</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
<td>478#</td>
<td>509#</td>
<td>524#</td>
<td>527#</td>
<td>531#</td>
<td>534#</td>
<td>540#</td>
<td>571#</td>
<td>586#</td>
</tr>
<tr>
<td>POLISH</td>
<td>421#</td>
<td>181#</td>
<td>1917#</td>
<td>1769#</td>
<td>1775#</td>
<td>1522#</td>
<td>1592#</td>
<td>1596#</td>
<td>1597#</td>
<td>1603#</td>
<td>1641#</td>
<td>1688#</td>
</tr>
<tr>
<td>POLLST</td>
<td>421#</td>
<td>181#</td>
<td>1917#</td>
<td>1769#</td>
<td>1775#</td>
<td>1522#</td>
<td>1592#</td>
<td>1596#</td>
<td>1597#</td>
<td>1603#</td>
<td>1641#</td>
<td>1688#</td>
</tr>
<tr>
<td>POINT13</td>
<td>469#</td>
<td>982#</td>
<td>2159#</td>
<td>2493#</td>
<td>2539#</td>
<td>2621#</td>
<td>2626#</td>
<td>176#</td>
<td>1769#</td>
<td>1795#</td>
<td>730#</td>
<td>719#</td>
</tr>
<tr>
<td>PPN</td>
<td>427#</td>
<td>2159#</td>
<td>2493#</td>
<td>2539#</td>
<td>2621#</td>
<td>2626#</td>
<td>176#</td>
<td>1769#</td>
<td>1795#</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
</tr>
<tr>
<td>PPNT2</td>
<td>176#</td>
<td>1769#</td>
<td>1795#</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
<td>478#</td>
<td>509#</td>
<td>524#</td>
<td>527#</td>
<td>531#</td>
<td>534#</td>
</tr>
<tr>
<td>SCHOFF</td>
<td>730#</td>
<td>719#</td>
<td>441#</td>
<td>478#</td>
<td>509#</td>
<td>524#</td>
<td>527#</td>
<td>531#</td>
<td>534#</td>
<td>540#</td>
<td>571#</td>
<td>586#</td>
</tr>
<tr>
<td>QCMOD</td>
<td>426#</td>
<td>2269#</td>
<td>413#</td>
<td>639#</td>
<td>661#</td>
<td>668#</td>
<td>669#</td>
<td>708#</td>
<td>717#</td>
<td>727#</td>
<td>738#</td>
<td>748#</td>
</tr>
</tbody>
</table>

Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>STDWD</td>
<td>423#</td>
<td>495</td>
<td>556</td>
<td>561</td>
<td>562</td>
<td>563</td>
<td>1829</td>
<td>1830</td>
<td>1831</td>
<td>1832</td>
<td>1891</td>
<td>2230</td>
<td>2231</td>
<td>2233</td>
<td></td>
<td></td>
<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>STORE</td>
<td>2235</td>
<td>2567</td>
<td>2570</td>
<td>2580</td>
<td>2581</td>
<td>2584</td>
<td>2586</td>
<td>2587</td>
<td>2590</td>
<td>2592</td>
<td>2594</td>
<td>2597</td>
<td>2599</td>
<td>2600</td>
<td>2601</td>
<td></td>
<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>STPNL</td>
<td>1433</td>
<td>1442</td>
<td>1492</td>
<td>1502</td>
<td>1519</td>
<td>1567</td>
<td>1590</td>
<td>1638#</td>
<td>1639#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STPNG</td>
<td>1436</td>
<td>1448</td>
<td>1455</td>
<td>1505</td>
<td>1526</td>
<td>1570</td>
<td>1596</td>
<td>1604</td>
<td>1616</td>
<td>1846#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STRTDAT</td>
<td>1979</td>
<td>2111#</td>
<td>2113</td>
<td>2128#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STRT1</td>
<td>1948</td>
<td>2111#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STRT2</td>
<td>1935</td>
<td>2113#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STRNUM</td>
<td>2258</td>
<td>2254</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>
<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>STRSET</td>
<td>659</td>
<td>690#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STSS2</td>
<td>789</td>
<td>715#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>STST1</td>
<td>1048</td>
<td>1152</td>
<td>1156</td>
<td>1934</td>
<td>1958</td>
<td>2132#</td>
<td>2352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>SUBPDP</td>
<td>422#</td>
<td>510</td>
<td>511</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>SVDPOS</td>
<td>419#</td>
<td>577</td>
<td>579</td>
<td>597</td>
<td>2415</td>
<td>2416</td>
<td>2417</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>SYMIX</td>
<td>420#</td>
<td>513</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TA0</td>
<td>426#</td>
<td>564</td>
<td>1029</td>
<td>1084</td>
<td>1172</td>
<td>1200</td>
<td>1949</td>
<td>1969</td>
<td>1975</td>
<td>2023</td>
<td>2034</td>
<td>2064</td>
<td>2167</td>
<td>2194</td>
<td></td>
<td></td>
<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>TAGPNT</td>
<td>420#</td>
<td>634</td>
<td>1965</td>
<td>1970</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TALREGC</td>
<td>420#</td>
<td>651</td>
<td>655</td>
<td>636</td>
<td>1966</td>
<td>1967</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TAGREP</td>
<td>420#</td>
<td>1972</td>
<td>1977</td>
<td>2028</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TBFCK</td>
<td>985</td>
<td>1017</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TECN</td>
<td>2272</td>
<td>2276#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TEST1</td>
<td>943</td>
<td>1023#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TEXT</td>
<td>643</td>
<td>732#</td>
<td>735</td>
<td>741</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TFRCE</td>
<td>1401</td>
<td>1405#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TIME</td>
<td>424a</td>
<td>499</td>
<td>2543</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TIMEUP</td>
<td>932</td>
<td>958#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TMLOP</td>
<td>765</td>
<td>783#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TMLOLP</td>
<td>765</td>
<td>770#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TMPMOD</td>
<td>1387</td>
<td>1398#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TMPNNT</td>
<td>1764</td>
<td>1771#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TNPST</td>
<td>701</td>
<td>712#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TNUM</td>
<td>425a</td>
<td>517</td>
<td>966</td>
<td>2555</td>
<td>2557</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TOLCHK</td>
<td>421#</td>
<td>903</td>
<td>1915</td>
<td>1926</td>
<td>1927</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TOLC</td>
<td>421a</td>
<td>903</td>
<td>1915</td>
<td>1926</td>
<td>1927</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TOOBIG</td>
<td>2312</td>
<td>2322#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TPOINT</td>
<td>461#</td>
<td>633</td>
<td>1964</td>
<td>1974</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TRMBL</td>
<td>418#</td>
<td>755</td>
<td>799</td>
<td>894</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TRNON</td>
<td>820</td>
<td>833</td>
<td>844#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TS1</td>
<td>2390#</td>
<td>2395</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TS2</td>
<td>2392</td>
<td>2396#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TS3</td>
<td>2398</td>
<td>2405#</td>
<td>2409</td>
<td>2413</td>
<td>2426</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TST5</td>
<td>2411</td>
<td>2432#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>TSTCMD</td>
<td>1058#</td>
<td>2362#</td>
<td>2436</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UANOD</td>
<td>2512</td>
<td>2604#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UCCMD</td>
<td>429a</td>
<td>499</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UFCMD</td>
<td>429a</td>
<td>2204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UNCA1</td>
<td>415</td>
<td>1982#</td>
<td>1986</td>
<td>2011</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UNCA2</td>
<td>1944</td>
<td>1962#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UNCD</td>
<td>1973a</td>
<td>1978</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UNFLOW</td>
<td>1241</td>
<td>1244#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>UPCMD</td>
<td>419a</td>
<td>2010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>USCHD</td>
<td>429a</td>
<td>2002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<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>
TITLE CALC

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT  COMPLETED 14-MAY-74

PART OF LEARN.SHR, VERSION 3

CALC.MAC 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 GI 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
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>CALC = FROM INTERP, MAC (SECTION H), LOCALED IN SECTION A</td>
</tr>
<tr>
<td>55</td>
<td>CALC = FROM LEARN, MAC (SECTION H), LOCALED IN SECTION A</td>
</tr>
<tr>
<td>56</td>
<td>CFRAME = FROM INTERP, MAC (SECTION B), LOCALED IN SECTION A</td>
</tr>
<tr>
<td>57</td>
<td>CFRAME = FROM INTERP, MAC (SECTIONS E &amp; K) AND ACTION, MAC (SECTION A, C, D, E, F, H &amp; I), LOCALED IN SECTION A</td>
</tr>
<tr>
<td>58</td>
<td>POLISH = FROM INTERP, MAC (SECTION Y) AND ACTION, MAC (SECTION E), LOCALED IN SECTION G</td>
</tr>
<tr>
<td>59</td>
<td>EXTERNAL SUBROUTINES CALLED BY CALC, MAC</td>
</tr>
<tr>
<td>60</td>
<td>BOMD2 = EXECUTES BRANCHING (B1), LOCALED IN ACTION, MAC (SECTION K)</td>
</tr>
<tr>
<td>61</td>
<td>OPROS = PROCESSES CALC MODE STATEMENTS, LOCALED IN OPROS, MAC (SECTION A)</td>
</tr>
<tr>
<td>62</td>
<td>FABS = COMPUTES ABSOLUTE VALUE OF NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION A)</td>
</tr>
<tr>
<td>63</td>
<td>FACOS = COMPUTES ARCCOS IN RADIANS OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION H)</td>
</tr>
<tr>
<td>64</td>
<td>FACOSD = COMPUTES ARCCOS IN DEGREES OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION P)</td>
</tr>
<tr>
<td>65</td>
<td>FASIN = COMPUTES THE ARCSIN IN RADIANS OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION H)</td>
</tr>
<tr>
<td>66</td>
<td>FASIND = COMPUTES ARCSIN IN DEGREES OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION Q)</td>
</tr>
<tr>
<td>67</td>
<td>FATAN = COMPUTES THE ARCTAN IN RADIANS OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION L)</td>
</tr>
<tr>
<td>68</td>
<td>FATAND = COMPUTES THE ARCTAN IN DEGREES OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION Q)</td>
</tr>
<tr>
<td>69</td>
<td>FCOS = COMPUTES THE COSINE OF THE ANGLE (RADIANS) IN AC-A, LOCALED IN FUNCS, MAC (SECTION J)</td>
</tr>
<tr>
<td>70</td>
<td>FCOSD = COMPUTES THE COSINE OF THE ANGLE (DEGREES) IN AC-A, LOCALED IN FUNCS, MAC (SECTION H)</td>
</tr>
<tr>
<td>71</td>
<td>FEXP = COMPUTES THE EXPONENTIAL OF THE NUMBER IN AC-A, LOCALED IN FUNCS, MAC (SECTION E)</td>
</tr>
<tr>
<td>72</td>
<td>FINT = TRUNCATES THE NUMBER IN AC-A TO AN INTEGER, LOCALED IN FUNCS, MAC (SECTION B)</td>
</tr>
<tr>
<td>Line</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<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>FNLOG = COMPUTES THE NATURAL LOGARITHM OF THE NUMBER IN AC-A, LOCATED IN FUNCS.MAC (SECTION G)</td>
</tr>
<tr>
<td>112</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>FRAN = PUTS A RANDOM FLOATING POINT NUMBER IN AC-A, LOCATED IN FUNCS.MAC (SECTION C)</td>
</tr>
<tr>
<td>115</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>FSIN = COMPUTES THE SINE OF THE ANGLE (RADIANS) IN AC-A, LOCATED IN FUNCS.MAC (SECTION K)</td>
</tr>
<tr>
<td>118</td>
<td></td>
</tr>
<tr>
<td>119</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>FSIND = COMPUTES THE SINE OF THE ANGLE (DEGREES) IN AC-A, LOCATED IN FUNCS.MAC (SECTION I)</td>
</tr>
<tr>
<td>121</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>FSQRT = COMPUTES THE SQUARE ROOT OF THE NUMBER IN AC-A, LOCATED IN FUNCS.MAC (SECTION D)</td>
</tr>
<tr>
<td>124</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td></td>
</tr>
<tr>
<td>126</td>
<td>INFLP = &quot;INPUTS&quot; A FLOATING POINT NO. INTO AC-B, LOCATED IN AUX1.MAC (SECTION F)</td>
</tr>
<tr>
<td>127</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>OUTFLP = &quot;OUTPUTS&quot; THE FLOATING POINT NO. IN AC-A, LOCATED IN AUX1.MAC (SECTION G)</td>
</tr>
<tr>
<td>130</td>
<td></td>
</tr>
<tr>
<td>131</td>
<td></td>
</tr>
<tr>
<td>132</td>
<td>RETSUB = RETURNS FROM A SUBROUTINE, LOCATED IN ACTION.MAC (SECTION J)</td>
</tr>
<tr>
<td>133</td>
<td></td>
</tr>
<tr>
<td>134</td>
<td></td>
</tr>
<tr>
<td>135</td>
<td>SKPTXT = SKIPS CHARACTERS IN A TEXT STATEMENT, LOCATED IN INTERP.MAC (SECTION AA)</td>
</tr>
<tr>
<td>136</td>
<td></td>
</tr>
<tr>
<td>137</td>
<td></td>
</tr>
<tr>
<td>138</td>
<td>TTYIN = &quot;INPUTS&quot; A FLOATING POINT NO. FROM A USER'S TERMINAL INTO AC-B, LOCATED IN AUX1.MAC (SECTION F)</td>
</tr>
<tr>
<td>139</td>
<td></td>
</tr>
<tr>
<td>140</td>
<td></td>
</tr>
<tr>
<td>141</td>
<td>EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOM1.MAC)</td>
</tr>
<tr>
<td>142</td>
<td></td>
</tr>
<tr>
<td>143</td>
<td>AC14 = CALL STATEMENT STORES THE BUFFER BYTE POINTER HERE FOR USE BY THE RETURN STATEMENT</td>
</tr>
<tr>
<td>144</td>
<td></td>
</tr>
<tr>
<td>145</td>
<td></td>
</tr>
<tr>
<td>146</td>
<td>AC15 = CALL STATEMENT STORES THE BUFFER BYTE COUNT HERE FOR USE BY THE RETURN STATEMENT</td>
</tr>
<tr>
<td>147</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>ANSTIM = INITIAL ANSWERING TIME, ACCUMULATED ANSWERING TIME, AND WAIT TIME (3 WORDS)</td>
</tr>
<tr>
<td>150</td>
<td></td>
</tr>
<tr>
<td>151</td>
<td></td>
</tr>
<tr>
<td>152</td>
<td>ARGNT = NUMBER OF ARGUMENT IN A SPECIAL FUNCTION</td>
</tr>
<tr>
<td>153</td>
<td></td>
</tr>
<tr>
<td>154</td>
<td>CBSAV = DISK BLOCK NO., BUFFER BYTE POINTER, AND BUFFER BYTE COUNT AT BEGINNING OF C-FRAME (3 WORDS)</td>
</tr>
<tr>
<td>155</td>
<td></td>
</tr>
<tr>
<td>156</td>
<td></td>
</tr>
<tr>
<td>157</td>
<td>DLCSTR = CALC STATEMENT BUFFER (16 WORDS)</td>
</tr>
<tr>
<td>158</td>
<td></td>
</tr>
<tr>
<td>159</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>CONST = CONSTANT STORAGE (64 WORDS)</td>
</tr>
<tr>
<td>161</td>
<td></td>
</tr>
</tbody>
</table>
DATA = STARTING ADDRESS FOR STORAGE OF STUDENT VARIABLES, LESSON VARIABLES, AND STUDENT PERFORMANCE DATA

DBLK = CALL STATEMENT STORES DISK BLOCK NO. HERE FOR USE BY RETURN STATEMENT

DEVICE = DEVICE NO. ASSOCIATED WITH USER'S TERMINAL

FIELD = TOTAL FIELD WIDTH AND NO. OFDIGITS AFTER DECIMAL POINT (USED BY FLOATING OUTPUT ROUTINE) (2 WORDS)

FRMTOT = TOTAL NO. OF FRAMES EXECUTED

IDXSAV = DATA INDEX IS STORED HERE SO THAT AC=F CAN BE USED

IDX1 = ADDRESS OF 1ST INDEX VARIABLE

IDXIV = FINAL VALUE OF 1ST INDEX

IDX2 = ADDRESS OF 2ND INDEX VARIABLE

IDX2I = INITIAL VALUE OF 2ND INDEX AND FINAL VALUE OF 2ND INDEX (2 WORDS)

IDX = INDEX TO THE 1ST WORD CONTAINING STUDENT PERFORMANCE DATA

INAM = DISK LOOKUP ARGUMENTS (4 WORDS)

LESHRD2 (#DIGITS) = NO. OF SPACES WHICH MUST BE TYPED AFTER A LEFT-JUSTIFIED NO.

LKSAY = LOOKUP ARGUMENTS FOR ORIGINAL LESSON (4 WORDS)

NUMU = NO. OF NEUTRAL ANSWERS

NUMNT = INDEX TO NUMBERS STORED IN THE CONSTANT LIST BY AN INPUT STATEMENT

PC3 = SUBROUTINES STORE RETURN ADDRESS HERE

PC4 = SUBROUTINES STORE RETURN ADDRESS HERE

POLLST = POLISH STRING PUSH DOWN LIST (64 WORDS)

RANNU = INTEGER RANDOM NO. USED BY RANDOM FUNCTION

RNU = NO. OF CORRECT ANSWERS

RSVNUM = TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE

SANS = STUDENT ANSWER BUFFER (16 WORDS)
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>213</td>
<td>SNANS = NUMERIC VALUE OF LAST STUDENT ANSWER</td>
</tr>
<tr>
<td>214</td>
<td>STORE (#DOSBLK) - CURRENT DISK BLOCK NO.</td>
</tr>
<tr>
<td>215</td>
<td>SUBPDP = PUSH DOWN LIST FOR STORING INFORMATION NEEDED TO RETURN</td>
</tr>
<tr>
<td>216</td>
<td>FROM A SUBROUTINE - 1ST WORD CONTAINS PUSH DOWN POINTER</td>
</tr>
<tr>
<td>217</td>
<td>(10 WORDS)</td>
</tr>
<tr>
<td>218</td>
<td>TIME = TIME OF DAY LESSON HAS STARTED</td>
</tr>
<tr>
<td>219</td>
<td>TNUM = NO. OF TIME UPS</td>
</tr>
<tr>
<td>220</td>
<td>TOL = NUMERIC ANSWER ERROR TOLERANCE</td>
</tr>
<tr>
<td>221</td>
<td>WNUM = NO. OF WRONG ANSWERS</td>
</tr>
<tr>
<td>222</td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>DESCRIPTION OF CFLAGS STORED IN AC=16</td>
</tr>
<tr>
<td>224</td>
<td>BIT 2 C-FRAME</td>
</tr>
<tr>
<td>225</td>
<td>BIT 3 IF STATEMENT</td>
</tr>
<tr>
<td>226</td>
<td>BIT 16 CALC MODE</td>
</tr>
<tr>
<td>227</td>
<td>BIT 17 COMMAND LEVEL</td>
</tr>
<tr>
<td>228</td>
<td>BIT 25 NEGATIVE NO.</td>
</tr>
<tr>
<td>229</td>
<td>BIT 30 NEGATIVE EXPONENT</td>
</tr>
<tr>
<td>230</td>
<td>BIT 33 2ND INDEX</td>
</tr>
<tr>
<td>231</td>
<td>BIT 34 INSIDE ITERATION COMPLETE</td>
</tr>
<tr>
<td>232</td>
<td>BIT 35 OUTSIDE ITERATION COMPLETE</td>
</tr>
<tr>
<td>233</td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>X</td>
</tr>
<tr>
<td>235</td>
<td>MISEG</td>
</tr>
<tr>
<td>236</td>
<td>ENTRY CALC,CFRAME,Polish,CALCH</td>
</tr>
<tr>
<td>237</td>
<td>EXTERN CPFRS,RANNUM,NUMPNT,EDO,FSGIN,CMON2,DEVICE,.Resub</td>
</tr>
<tr>
<td>238</td>
<td>EXTERN FATAN,FSIN,FACOS,FATTAN,FSIND,FACOSD,IDXAV,SNANS,SNAN</td>
</tr>
<tr>
<td>239</td>
<td>EXTERN FPXR,FDLOG,FDLOG,PCOS,FCOS,FSIN,FSIND,TLR,ARGCNT,SKPTXT</td>
</tr>
<tr>
<td>240</td>
<td>EXTERN RNUN,NNUM,NNUM,TNUM,TIME,ANSTM,SUBPDP,TTYIN;LESRO;LMSAV</td>
</tr>
<tr>
<td>241</td>
<td>EXTERN FIELD,OFLP,FINT,FADS,FRAN,FRXROT,FSORT,IDX</td>
</tr>
<tr>
<td>242</td>
<td>EXTERN CBSCAV,DATA,STORE,IDXV,MDTX,PC4,PC4,DOSBLK,INAH</td>
</tr>
<tr>
<td>243</td>
<td>EXTERN IDX2,IDX2R,RSNUM,CLCSTR,CONST,POLLST,INFLP,AC14,AC15</td>
</tr>
<tr>
<td>244</td>
<td></td>
</tr>
<tr>
<td>245</td>
<td>000001</td>
</tr>
<tr>
<td>246</td>
<td>000002</td>
</tr>
<tr>
<td>247</td>
<td>000003</td>
</tr>
<tr>
<td>248</td>
<td>000004</td>
</tr>
<tr>
<td>249</td>
<td>000005</td>
</tr>
<tr>
<td>250</td>
<td>000006</td>
</tr>
<tr>
<td>251</td>
<td>000007</td>
</tr>
<tr>
<td>252</td>
<td>000008</td>
</tr>
<tr>
<td>253</td>
<td>000009</td>
</tr>
<tr>
<td>254</td>
<td>000010</td>
</tr>
<tr>
<td>255</td>
<td>000011</td>
</tr>
<tr>
<td>256</td>
<td>000012</td>
</tr>
<tr>
<td>257</td>
<td>000013</td>
</tr>
<tr>
<td>258</td>
<td>000014</td>
</tr>
<tr>
<td>259</td>
<td>000015</td>
</tr>
<tr>
<td>260</td>
<td>000016</td>
</tr>
<tr>
<td>261</td>
<td>000017</td>
</tr>
<tr>
<td>262</td>
<td>000018</td>
</tr>
<tr>
<td>263</td>
<td>000019</td>
</tr>
<tr>
<td>264</td>
<td>000020</td>
</tr>
<tr>
<td>265</td>
<td>000021</td>
</tr>
<tr>
<td>266</td>
<td>000022</td>
</tr>
<tr>
<td>267</td>
<td>000023</td>
</tr>
<tr>
<td>268</td>
<td>000024</td>
</tr>
<tr>
<td>269</td>
<td>000025</td>
</tr>
<tr>
<td>270</td>
<td>000026</td>
</tr>
<tr>
<td>271</td>
<td>000027</td>
</tr>
<tr>
<td>272</td>
<td>000028</td>
</tr>
<tr>
<td>273</td>
<td>000029</td>
</tr>
<tr>
<td>274</td>
<td>000030</td>
</tr>
<tr>
<td>275</td>
<td>000031</td>
</tr>
<tr>
<td>276</td>
<td>000032</td>
</tr>
<tr>
<td>277</td>
<td>000033</td>
</tr>
<tr>
<td>278</td>
<td>000034</td>
</tr>
<tr>
<td>279</td>
<td>000035</td>
</tr>
<tr>
<td>280</td>
<td>000036</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION B. PROCESSES ITERATION FIELDS.

ITER I TRE OFLAGS, 7 I CLEAR THE ITERATION FLAGS

ITER I JSP 0, 11 I INPUT <02>

ITER I JSP 0, 11 I INPUT 1ST DIGIT OF VARIABLE ADDRESS

MOVE B, 1, 4B (A) I PUT VALUE OF DIGIT IN AC-B

INPUT B, 1, 0B I SHIFT 1 DIGIT LEFT (BASE 00)

JSP 0, 11 I PUT 2ND DIGIT OF ADDRESS

ADDI B, 40 (A) I ADD DIGIT TO NO. IN AC-B

MOVE A, RSVNUM I PUT NO. OF WORDS OF STUDENT MEMORY IN AC-A

ADDI B, DATA (A) I ADD ON BASE ADDRESS OF LESSON MEMORY

TRNE OFLAGS, 4 I 2ND INDEX VARIABLE?

JST ITER IS YES, GO TO ITER (SECTION B)

JST ITER IS ELSE, SAVE ADDRESS OF INDEX VARIABLE IN IDX1

JST ITER IS MOVE B, IDX1

JST ITER IS MOVE B, IDX2 I SAVE ADDRESS OF 2ND INDEX VARIABLE IN IDX2

JST ITER IS MOVE G, POINT 1 I INPUT POINTER IN AC-G

JSP 0, 11 I INPUT CHARACTER

IDPB A, G I PUT IT IN CALC STATEMENT BUFFER

CAIE A, 1, 1 I END OF EXPRESSION?

JST -3 I NO, GO INPUT NEXT CHARACTER

MOVE G, POINT 1 I RESET BYTE POINTER

JSP 0, 10 I EVALUATE THE EXPRESSION IN THE BUFFER

MOVE B, POLLST I PUT VALUE OF EXPRESSION IN AC-B

TRNE OFLAGS, 4 I 2ND INDEX?

JST ITER IS YES, GO TO ITER

JST ITER IS MOVE B, IDX1 I SET INDEX VARIABLE TO INITIAL VALUE

JST ITER IS
I SECTION C. LOADS CALC STATEMENT INTO STATEMENT BUFFER.

LOAD: MOVE C,P O N P I N C A L C STATEMENT BUFFER BYTE POINTER IN AC = 0

I SECTION D. INCREMENTS ITERATION INDICES.

ITERATI TRUE CFLAGS = 1 IS THERE A 2ND INDEX VARIABLE?
CALC MACRO 47(204)-7 00113 2-AUG-74 PAGE 1-16

CALC MAC 8-JUL-74

<table>
<thead>
<tr>
<th>MOV G.POINT1</th>
<th>RESET CALC BUFFER BYTE POINTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSP G.111</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>IDPE A.D</td>
<td>INPUT IT IN CALC STATEMENT BUFFER</td>
</tr>
<tr>
<td>CAIE A.15</td>
<td>END OF LINE?</td>
</tr>
<tr>
<td>JRST -3</td>
<td>NO, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>MOV G.POINT1</td>
<td>RESET BYTE POINTER</td>
</tr>
<tr>
<td>FL61 TRE CFLAGS2008 ISENTER NEGATIVE NO. FLAG</td>
<td></td>
</tr>
<tr>
<td>IDLB A.B</td>
<td>GET CHARACTER FROM CALC STATEMENT BUFFER</td>
</tr>
<tr>
<td>CAIE A.&quot;&quot;</td>
<td>MINUS SIGN?</td>
</tr>
<tr>
<td>JRST -3</td>
<td>NO, JUMP</td>
</tr>
<tr>
<td>MOV G.POINT1</td>
<td>ELSE, PUT NEW BYTE POINTER IN AC=G</td>
</tr>
<tr>
<td>TRO CFLAGS2008 ISENTER NEGATIVE NO. FLAG</td>
<td></td>
</tr>
<tr>
<td>JSP Q.INFLP</td>
<td>&quot;INPUT&quot; NO.</td>
</tr>
<tr>
<td>TRNE CFLAGS2008 ISENTER NO. NEGATIVE</td>
<td></td>
</tr>
<tr>
<td>MOVNS B IYES</td>
<td>NEGATE IT</td>
</tr>
<tr>
<td>MOVIE A.IIDX</td>
<td>GET INITIAL DATA INDEX</td>
</tr>
<tr>
<td>IDPB C.DEP(0)</td>
<td>GET 1ST ADDRESS OF STUDENT PERF. DATA IN AC=G</td>
</tr>
<tr>
<td>CAMG C.VALUE</td>
<td>WILL NO BE STORED IN PROPER AREA</td>
</tr>
<tr>
<td>JRST MERRY</td>
<td>NO, GO INPUT MESSAGE (SECTION F)</td>
</tr>
<tr>
<td>MOVIE B.IIDX</td>
<td>STORE THE NO.</td>
</tr>
<tr>
<td>AOS VALUE</td>
<td>INCREMENT ARRAY ADDRESS</td>
</tr>
<tr>
<td>SOS VALUE</td>
<td>IDENTITY NO. OF NUMBERS TO INPUT</td>
</tr>
<tr>
<td>CAIE A.&quot;&quot;</td>
<td>END OF LINE OF NOS?</td>
</tr>
<tr>
<td>JRST FL6</td>
<td>NO, GO TO FL6</td>
</tr>
<tr>
<td>JRST FL9</td>
<td>YES, GO TO FL5</td>
</tr>
<tr>
<td>STRFILL C.POINT7.03</td>
<td>GET 1ST PART OF BYTE POINTER</td>
</tr>
<tr>
<td>HRR A.77</td>
<td>ADD-ON ADDRESS PORTION OF THE BYTE POINTER</td>
</tr>
<tr>
<td>IDLB A.B</td>
<td>GET CHARACTER FROM CALC STATEMENT BUFFER</td>
</tr>
<tr>
<td>CAIE A.&quot;&quot;</td>
<td>END OF STRING?</td>
</tr>
<tr>
<td>JRST ITERAT</td>
<td>YES, GO TO ITERAT</td>
</tr>
<tr>
<td>IDPB C.155</td>
<td>ELSE, STORE THE CHARACTER</td>
</tr>
<tr>
<td>MOVIE B.IIDX</td>
<td>GET INITIAL DATA INDEX</td>
</tr>
<tr>
<td>MOVIE B.DATA(S)</td>
<td>GET INITIAL DATA LOCATION</td>
</tr>
<tr>
<td>CAIE B.C)</td>
<td>WAS CHARACTER STORED IN DATA AREA</td>
</tr>
<tr>
<td>JRST MERRY</td>
<td>YES, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>JRST -15</td>
<td>NO, GO NEXT CHARACTER</td>
</tr>
</tbody>
</table>

SECTION H. EXECUTES TYPE STATEMENTS,

<table>
<thead>
<tr>
<th>CTYPER</th>
<th>MOVIE G.G</th>
<th>PUT BYTE POINTER IN AC=G</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLB A.G</td>
<td>GET CHARACTER FROM CALC STATEMENT BUFFER</td>
<td></td>
</tr>
<tr>
<td>CAIE A.&quot;&quot;</td>
<td>NOT CHARACTER A COMMA</td>
<td></td>
</tr>
<tr>
<td>JRST CTYPE</td>
<td>YES, GO INPUT NEXT CHARACTER</td>
<td></td>
</tr>
<tr>
<td>JRST CTYPE</td>
<td>YES, GO INPUT TEXT EXPRESSION</td>
<td></td>
</tr>
<tr>
<td>JRST CTYPE</td>
<td>YES, GO TO TEXT TEXT</td>
<td></td>
</tr>
<tr>
<td>JRST -15</td>
<td>IF END OF STATEMENT OR TEXT_EXPRESSION, GO TO ITERAT (SECTION D)</td>
<td></td>
</tr>
<tr>
<td>JRST ITERAT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R0022 200 08 06 080194 |
| MOVIE G.C | ELSE, PUT BYTE POINTER IN AC=G |
SECTION N. EXECUTES ASSIGNMENT STATEMENTS:

ASSIGN MOVE G,POINT SET BYTE POINTER TO BEGINNING OF STATEMENT
LINE CLFOLS,40000 IS THIS AN IF STATEMENT?
MOVE G,B SET BYTE POINTER TO ASSIGNMENT STATEMENT
JSP G,POLISH EXECUTE THE STATEMENT
JST ITERAT 1G0 TO ITERAT (SECTION D)

SECTION O. EVALUATES POLISH STRINGS. AS THE STRING IS SCANNED,
OPERAND ADDRESSES ARE PLACED IN A PUSH DOWN LIST (CONSTANTS ARE
PLACED IN THE CONSTANT LIST, WHEN AN OPERATOR IS ENCOUNTERED,
THE LAST TWO ADDRESSES ARE POPPED OUT OF THE PUSH DOWN LIST,
AND THE OPERATION IS CARRIED OUT ON THE TWO NUMBERS FOUND AT THESE
ADDRESSES. IN THE CASE OF UNARY MINUS OR A FUNCTION, ONLY ONE
ADDRESS IS POPPED OUT OF THE PUSH DOWN LIST. THE RESULT OF THE
OPERATION IS THEN PLACED IN THE CONSTANT LIST, AND ITS ADDRESS IN
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.

```
1000 MOVE Q,PC4:ISAVE RETURN ADDRESS
1001 MOVEM D,PO:LIST=1:ISET UP PUSH DOWN POINTER
1010
1011 ILDB A:G:ICALL CHARACTER FROM CALC STATEMENT BUFFER
1012
1013 001163 202 17 0 00 000000
1014 001164 261 04 0 00 77777
1015 001169 400 16 0 00 0000
1016 001166 134 01 0 00 00007
1017 001167 301 01 0 00 00016
1018 001167 254 00 0 00 001005
1019 001167 254 00 0 00 001005
1020 001167 254 00 0 00 001005
1021 001167 254 00 0 00 001005
1022 001167 254 00 0 00 001005
1023 001167 254 00 0 00 001005
1024 001167 254 00 0 00 001005
1025 001167 254 00 0 00 001005
1026 001167 254 00 0 00 001005
1027 001167 254 00 0 00 001005
1028 001167 254 00 0 00 001005
1029 001167 254 00 0 00 001005
1030 001167 254 00 0 00 001005
1031 001167 254 00 0 00 001005
1032 001167 254 00 0 00 001005
1033 001167 254 00 0 00 001005
1034 001167 254 00 0 00 001005
1035 001167 254 00 0 00 001005
1036 001167 254 00 0 00 001005
1037 001167 254 00 0 00 001005
1038 001167 254 00 0 00 001005
1039 001167 254 00 0 00 001005
1040 001167 254 00 0 00 001005
1041 001167 254 00 0 00 001005
1042 001167 254 00 0 00 001005
1043 001167 254 00 0 00 001005
1044 001167 254 00 0 00 001005
1045 001167 254 00 0 00 001005
1046 001167 254 00 0 00 001005
1047 001167 254 00 0 00 001005
1048 001167 254 00 0 00 001005
1049 001167 254 00 0 00 001005
1050 001167 254 00 0 00 001005
1051 001167 254 00 0 00 001005
1052 001167 254 00 0 00 001005
1053 001167 254 00 0 00 001005
1054 001167 254 00 0 00 001005
1055 001167 254 00 0 00 001005
1056 001167 254 00 0 00 001005
1057 001167 254 00 0 00 001005
1058 001167 254 00 0 00 001005
1059 001167 254 00 0 00 001005
1060 001167 254 00 0 00 001005
1061 001167 254 00 0 00 001005
1062 001167 254 00 0 00 001005
1063 001167 254 00 0 00 001005
1064 001167 254 00 0 00 001005
1065 001167 254 00 0 00 001005
1066 001167 254 00 0 00 001005
1067 001167 254 00 0 00 001005
1068 001167 254 00 0 00 001005
```
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1114</td>
<td>001321</td>
<td>254 00 00 00 002153</td>
<td>JRST MINUS YES, GO TO MINUS</td>
</tr>
<tr>
<td>1115</td>
<td>001322</td>
<td>306 01 00 00 00654</td>
<td>CAIN A, &quot;&quot; COMMA</td>
</tr>
<tr>
<td>1116</td>
<td>001323</td>
<td>254 00 00 00 002271</td>
<td>JRST COMMA YES, GO TO COMMA</td>
</tr>
<tr>
<td>1117</td>
<td>001324</td>
<td>306 01 00 00 00137</td>
<td>CAIN A, &quot;&quot; REPLACEMENT</td>
</tr>
<tr>
<td>1118</td>
<td>001325</td>
<td>254 00 00 00 002156</td>
<td>JRST REPLACE YES, GO TO REPLACE</td>
</tr>
<tr>
<td>1119</td>
<td>001326</td>
<td>306 01 00 00 00675</td>
<td>CAIN A, &quot;&quot; EQUALS</td>
</tr>
<tr>
<td>1120</td>
<td>001327</td>
<td>254 00 00 00 002221</td>
<td>JRST EQUALS YES, GO TO EQUALS</td>
</tr>
<tr>
<td>1121</td>
<td>001328</td>
<td>306 01 00 00 00774</td>
<td>CAIN A, &quot;&quot; LESS THAN</td>
</tr>
<tr>
<td>1122</td>
<td>001329</td>
<td>254 00 00 00 0022101</td>
<td>JRST LESS YES, GO TO LESS</td>
</tr>
<tr>
<td>1123</td>
<td>001330</td>
<td>306 01 00 00 00676</td>
<td>CAIN A, &quot;&quot; GREATER THAN</td>
</tr>
<tr>
<td>1124</td>
<td>001331</td>
<td>254 00 00 00 002217</td>
<td>JRST MORE YES, GO TO MORE</td>
</tr>
<tr>
<td>1125</td>
<td>001332</td>
<td>306 01 00 00 00682</td>
<td>CAIN A,61 INST EQUAL</td>
</tr>
<tr>
<td>1126</td>
<td>001333</td>
<td>254 00 00 00 002258</td>
<td>JRST NOTEQ YES, GO TO NOTEQ</td>
</tr>
<tr>
<td>1127</td>
<td>001334</td>
<td>306 01 00 00 00662</td>
<td>CAIN A,62 LESS THAN OR EQUAL</td>
</tr>
<tr>
<td>1128</td>
<td>001335</td>
<td>254 00 00 00 002235</td>
<td>JRST LESEQ YES, GO TO LESEQ</td>
</tr>
<tr>
<td>1129</td>
<td>001336</td>
<td>306 01 00 00 00933</td>
<td>CAIN A,63 GREATER THAN OR EQUAL</td>
</tr>
<tr>
<td>1130</td>
<td>001337</td>
<td>254 00 00 00 002244</td>
<td>JRST GRTEQ YES, GO TO GRTEQ</td>
</tr>
<tr>
<td>1131</td>
<td>001338</td>
<td>306 01 00 00 00856</td>
<td>CAIN A,65 K65?</td>
</tr>
<tr>
<td>1132</td>
<td>001339</td>
<td>254 00 00 00 001166</td>
<td>JRST POL1 YES, GO TO POL1</td>
</tr>
<tr>
<td>1133</td>
<td>001340</td>
<td>254 00 01 00 001244</td>
<td>JRST W=180(A) ELSE, JUMP TO ROUTINE TO EXECUTE FUNCTION</td>
</tr>
<tr>
<td>1134</td>
<td>001341</td>
<td>000000 0010000013771</td>
<td>XHO B,SRT</td>
</tr>
<tr>
<td>1135</td>
<td>001342</td>
<td>000000 0014931</td>
<td>XHO B,ABS</td>
</tr>
<tr>
<td>1136</td>
<td>001343</td>
<td>000000 0014877</td>
<td>XHO B,INT</td>
</tr>
<tr>
<td>1137</td>
<td>001344</td>
<td>000000 001414</td>
<td>XHO B,CRT</td>
</tr>
<tr>
<td>1138</td>
<td>001345</td>
<td>000000 001426</td>
<td>XHO B,EXP</td>
</tr>
<tr>
<td>1139</td>
<td>001346</td>
<td>000000 0014321</td>
<td>XHO B,NLOG</td>
</tr>
<tr>
<td>1140</td>
<td>001347</td>
<td>000000 001436</td>
<td>XHO B,LOG</td>
</tr>
<tr>
<td>1141</td>
<td>001348</td>
<td>000000 0014164</td>
<td>XHO B,TAN</td>
</tr>
<tr>
<td>1142</td>
<td>001349</td>
<td>000000 001442</td>
<td>XHO B,TAND</td>
</tr>
<tr>
<td>1143</td>
<td>001350</td>
<td>000000 0014771</td>
<td>XHO B,SIN</td>
</tr>
<tr>
<td>1144</td>
<td>001351</td>
<td>000000 001533</td>
<td>XHO B,SIND</td>
</tr>
<tr>
<td>1145</td>
<td>001352</td>
<td>000000 001507</td>
<td>XHO B,COS</td>
</tr>
<tr>
<td>1146</td>
<td>001353</td>
<td>000000 001513</td>
<td>XHO B,COSB</td>
</tr>
<tr>
<td>1147</td>
<td>001354</td>
<td>000000 001523</td>
<td>XHO B,ACOS</td>
</tr>
<tr>
<td>1148</td>
<td>001355</td>
<td>000000 001517</td>
<td>XHO B,ASIN</td>
</tr>
<tr>
<td>1149</td>
<td>001356</td>
<td>000000 001527</td>
<td>XHO B,ATAN</td>
</tr>
<tr>
<td>1150</td>
<td>001357</td>
<td>000000 001533</td>
<td>XHO B,ACOSD</td>
</tr>
<tr>
<td>1151</td>
<td>001358</td>
<td>000000 001577</td>
<td>XHO B,ASIND</td>
</tr>
<tr>
<td>1152</td>
<td>001359</td>
<td>000000 001543</td>
<td>XHO B,ATAN</td>
</tr>
<tr>
<td>1153</td>
<td>001360</td>
<td>000000 0016021</td>
<td>XHO B,RIGHT</td>
</tr>
<tr>
<td>1154</td>
<td>001361</td>
<td>000000 0016051</td>
<td>XHO B,WONG</td>
</tr>
<tr>
<td>1155</td>
<td>001362</td>
<td>000000 0016180</td>
<td>XHO B,NEUT</td>
</tr>
<tr>
<td>1156</td>
<td>001363</td>
<td>000000 001613</td>
<td>XHO B,SEEN</td>
</tr>
<tr>
<td>1157</td>
<td>001364</td>
<td>000000 0016143</td>
<td>XHO B,PATH</td>
</tr>
<tr>
<td>1158</td>
<td>001365</td>
<td>000000 001713</td>
<td>XHO B,ATIME</td>
</tr>
<tr>
<td>1159</td>
<td>001366</td>
<td>000000 001741</td>
<td>XHO B,TAG</td>
</tr>
<tr>
<td>1160</td>
<td>001367</td>
<td>262 04 00 00 002002</td>
<td>SORT1 PDP D,B IGET ADDRESS FROM PUSH DOWN LIST</td>
</tr>
<tr>
<td>1161</td>
<td>001481</td>
<td>262 01 00 00 009800</td>
<td>MOVE A, (B) IGET NO. FROM THAT LOCATION</td>
</tr>
<tr>
<td>1162</td>
<td>001482</td>
<td>254 17 00 00 002521</td>
<td>JSP Q,PSORT1 ITAKE SQUARE ROOT OF THE NO.</td>
</tr>
<tr>
<td>1163</td>
<td>001483</td>
<td>254 00 00 00 001252</td>
<td>JRST PUSH+1 IGO TO PUSH+1</td>
</tr>
<tr>
<td>1164</td>
<td>001484</td>
<td>262 04 00 00 005002</td>
<td>ABS1 PDP D,B IGET ADDRESS FROM PUSH DOWN LIST</td>
</tr>
<tr>
<td>1165</td>
<td>001485</td>
<td>262 01 00 00 009800</td>
<td>MOVE A, (B) IGET NUMBER FROM THAT LOCATION</td>
</tr>
<tr>
<td>1166</td>
<td>001486</td>
<td>254 17 00 00 009800</td>
<td>JSP Q,PSBS ITAKE ABSOLUTE VALUE OF THE NO.</td>
</tr>
</tbody>
</table>
ATANDI POP D, B IGET ADDRESS FROM PUSH DOWN LIST
MOVE A, (B) IGET NO. FROM THAT LOCATION
ISP D, FATAND IPOP D, ITAN OF THE NO.
JSTR 256-42, 128 TO PUSH+4
BP1  POINT 11, DATA (C), 20
BP2  POINT 7, DATA (C), 17
BP3  POINT 10, DATA (C), 23
BP4  POINT 7, DATA (C), 6
RWN1 JSP 0, SETARG IPOP UP FUNCTION ARGUMENTS
SETAB 1, F
MOVE C, IDX IGET INITIAL DATA INDEX
RWN1 CAML C, IDXSAY IEND OF STUDENT PERFORMANCE DATA?
JSTR ALL YES, GO TO SALL
LDAC, B, 8 IGET FRAME NO.
AOS C
RWN2 CAML C, IDXSAY IEND OF STUDENT PERFORMANCE DATA?
JSTR SALL YES, GO TO SALL
LDAC, B, 8 IELSE GET FRAME NO.
AOS C
RWN1 CAML C, IDXSAY IEND OF STUDENT PERFORMANCE DATA?
JSTR SALL YES, GO TO SALL
LDAC, B, 8 IELSE GET FRAME NO.
AOS C
RWN1 CAML C, IDXSAY IEND OF STUDENT PERFORMANCE DATA?
JSTR SALL YES, GO TO SALL
LDAC, B, 8 IELSE GET FRAME NO.
AOS C
RWN1 CAML C, IDXSAY IEND OF STUDENT PERFORMANCE DATA?
JSTR SALL YES, GO TO SALL
LDAC, B, 8 IELSE GET FRAME NO.
AOS C
TITLE ACTION

COMMENT %

WRITTEN BY GARREY A, VANDER LUGT       COMPLETED 18-MAY-74

PART OF LEARN, SHR, VERSION 3

ACTION, MAC EXECUTES THE ACTIONS FOLLOWING ACTION STATEMENTS IN G-FRAMES.

CONTENTS OF ACTION, MAC

SECTION A, EXECUTES FI STATEMENTS,

SECTION B, SUBROUTINE WHICH TYPES RANDOM COMMENTS,

SECTION C, EXECUTES CI STATEMENTS,

SECTION D, EXECUTES RI STATEMENTS,

SECTION E, EXECUTESAI STATEMENTS,

SECTION F, EXECUTES GI STATEMENTS,

SECTION G, EXECUTES SI STATEMENTS,

SECTION H, EXECUTES EI STATEMENTS,

SECTION I, EXECUTES LI STATEMENTS,

SECTION J, EXECUTES GI STATEMENTS,

SECTION K, EXECUTES BI STATEMENTS,

SECTION L, EXECUTES PI STATEMENTS.

ENTRY POINTS TO ACTION, MAC

ACMD = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION E

BCMD = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION K

BCMD2 = FROM INTERP,MAC (SECTION FF) AND CALC,MAC (SECTION I),

LOCATED IN SECTION K

CCMD = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION C

ECMD = FROM INTERP,MAC (SECTION 2), LOCATED IN SECTION M.
ACTION MACRO 47(204)-7 0812 2-AUG-74 PAGE 1-1

ACTION MAC 17-MAY-74

FCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION A
GCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION A
LCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION I
PCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION L
QCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION F
RCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION D
RETSUB = FROM INTERP,MAC (SECTION E) AND CALC,MAC (SECTION J), LOCATED IN SECTION J
SCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION G
UCCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION G
UFCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION A
UPCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION L
USCHD = FROM INTERP,MAC (SECTION E), LOCATED IN SECTION G

EXTERNAL SUBROUTINES CALLED BY ACTION,MAC

CFRAME = EXECUTES C-FRAME STATEMENTS, LOCATED IN CALC,MAC (SECTION A)

EXPCOR = EXPANDS THE LOW SEGMENT IF NECESSARY, LOCATED IN AUX1,MAC (SECTION H)
IID = INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX1,MAC (SECTION E)
OUTLP = OUTPUTS THE FLOATING POINT NO, IN AC-A, LOCATED IN AUX1,MAC (SECTION O)
PAG = PAGES CRT TERMINALS, LOCATED IN INTERP,MAC (SECTION F)
POLISH = EVALUATES POLISH EXPRESSIONS, LOCATED IN CALC,MAC (SECTION O)
SKPTXT = SKIPS CHARACTERS IN A TEXT STATEMENT, LOCATED IN INTERP,MAC (SECTION AA)
SLDRTN = OPERATES THE SLIDE PROJECTOR INTERFACE, LOCATED IN INTERP,MAC (SECTION F)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOW1,MAC)
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

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

FIELD = TOTAL FIELD WIDTH (USED BY FLOATING OUTPUT ROUTINE)

FRMNUM = CURRENT FRAME NO.

FRMTOT = TOTAL NO. OF FRAMES EXECUTED

FRMTYP = FRAME TYPE OF CURRENT FRAME

IBUF = DISK INPUT BUFFER HEADER (3 WORDS)

IDXSAV = DATA INDEX IS SAVED HERE WHEN AC-F IS USED BY A SUBPROGRAM OTHER THAN INTERP, MAC

POLLST = POLISH STRING PUSH DOWN LIST (64 WORDS)

RGTANS = CORRECT ANSWER BUFFER (16 WORDS)

STORE (#D KBK) = CURRENT DISK BLOCK NO.

SUBPDP = PUSH DOWN LIST FOR STORING INFORMATION NEEDED TO RETURN FROM A SUBROUTINE — 1ST WORD CONTAINS PUSH DOWN POINTER (16 WORDS)

SYPOS = INFORMATION NEEDED TO REPEAT QUESTION (Q) — DISK BLOCK NO., BUFFER BYTE POINTER, AND BUFFER BYTE COUNT (3 WORDS)

TAG = TAG OF THE LAST ANSWER

DESCRIPTION OF FLAGS STORED IN AC=8

BIT 3 CORRECT ANSWER

BIT 4 WRONG ANSWER

BIT 20 REVIEW IN PROGRESS

DESCRIPTION OF FLAGS STORED IN AC=16

BIT 2 Q-FRAME

% MISEC

ENTRY ACMD
SECTION B. SUBROUTINE TO GENERATE RANDOM COMMENTS.

1.0046 200 01 0 0 0 000000  MOV A,COMMUN
000003  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
1.0047 271 01 0 0 000115  ADDI A, 115
000004  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000005  ADD 77
1.0050 231 01 0 0 000144  IDIV A, 144
000006  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000007  DIVIDE BY 100
1.0051 200 02 0 0 000045  MOV B,COMMUN
000008  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000009  SAVE THE REMAINDER
1.0052 211 02 0 0 000012  IDIV B, 12
000000  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000001  DIVIDE REMAINDER BY 10
1.0054 254 01 0 0 000021  TLNN FLGS,60000
000002  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000003  INVESTR ANSWER?
1.0055 254 00 0 0 000025  JRT NEUT
000004  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000005  YES, GO TO NEUT
1.0056 254 01 0 0 000029  TLNE FLGS,20000
000006  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000007  YES, GO TO NEUT
1.0057 254 00 0 0 000033  JRT WRONG
000008  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000009  YES, GO TO WRONG
1.0058 254 00 0 0 000037  JRT Q
000000  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000001  RETURN
1.0060 254 00 0 0 000041  JRT Q
000002  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000003  RETURN
1.0061 003000 000000 000373  RCOMI XWD RCOM1
000004  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000005  XWD RCOM2
000006  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000007  XWD RCOM3
000008  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000009  XWD RCOM4
000010  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000011  XWD RCOM5
000012  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000013  XWD RCOM6
000014  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000015  XWD RCOM7
000016  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
000017  XWD RCOM8
000018  RANCOM MOVE A,COMMUN  GET COMMENT RANDOM NO.
SECTION D. INTERPRETS AND EXECUTES REPEAT (R/) STATEMENTS.

360 RCMD1 JSP 0.11 INPUT CHARACTER
361 RCM2 JSP 0.12 END OF LINE?
362 RCM3 JSP 0.13 YES, GO TO STDOCM
363 RCM4 JSP 0.14 TYPE TEXT
364 RCM5 JSP 0.15 TDCALL 3, CRLFO
365 RCM6 JSP 0.16 JSP 0.17
366 RCM7 JSP 0.18
367 RCM8 JSP 0.19
368 RCM9 JSP 0.20
369 RCM10 JSP 0.21
370 RCM11 JSP 0.22
371 RCM12 JSP 0.23
372 RCM13 JSP 0.24
373 RCM14 JSP 0.25
374 RCM15 JSP 0.26
375 RCM16 JSP 0.27
376 RCM17 JSP 0.28
377 RCM18 JSP 0.29
378 RCM19 JSP 0.30
379 RCM20 JSP 0.31
380 RCM21 JSP 0.32
381 RCM22 JSP 0.33
382 RCM23 JSP 0.34
383 RCM24 JSP 0.35
384 RCM25 JSP 0.36
385 RCM26 JSP 0.37
386 RCM27 JSP 0.38
387 RCM28 JSP 0.39
388 RCM29 JSP 0.40
389 RCM30 JSP 0.41
390 RCM31 JSP 0.42
391 RCM32 JSP 0.43
392 RCM33 JSP 0.44
393 RCM34 JSP 0.45
394 RCM35 JSP 0.46
395 RCM36 JSP 0.47
396 RCM37 JSP 0.48
397 RCM38 JSP 0.49
398 RCM39 JSP 0.50
399 RCM40 JSP 0.51
400 RCM41 JSP 0.52
401 RCM42 JSP 0.53
402 RCM43 JSP 0.54

TRY AGAIN.

/1 JSP 0.55
/2 JSP 0.56
/3 JSP 0.57
/4 JSP 0.58
/5 JSP 0.59

NO SUCH CHOICE, CHOOSE ONE OF THE ABOVE LETTERS.

REPEAT; MOVE A BLOCK, GET DISK BLOCK CONTAINING ANSWER STATEMENT

CAME A, DSKBLK IS SAME AS CURRENT DISK BLOCK?

USEST 1, @BLOCK NO, DO USEST?

CAME A, DSKBLK IS SAME AS CURRENT BLOCK?

INPUT 1, NO, INPUT BLOCK

MOVEN A, DSKBLK ISAVE THE NEW DISK BLOCK NO.

MOVE 19, BLOCK-2 INPUT BYTE COUNT IN AC-15

AOS INDEX INCREMENT DATA INDEX

JSP G, EXPGR IF NECESSARY

AOS FRMTOT ADD ONE TO FRAME COUNT
1. SECTION F: INTERPRETS AND EXECUTES Q1 STATEMENTS.

2. QCMD1  JSP Q.11  INPUT CHARACTER
3. CAIN A.19  END OF LINE?
4. JST STOCH  YES, GO TO STOCH
5. TPTXT  ^ TYPE TEXT

6. TCALC 3. CRLF0  JSP Q.11
7. JUMPE A.1.3
8. TCALL 1. A

9. JST R.3
10. JSP Q.11
11. CAIN A.15
12. JST R.44
13. TLO 16:10000
14. JSP Q.FRAME+1
15. STOCH  TLNN FLG6.60000  IS THIS A NEUTRAL ANSWER?
16. JST QCOM2  YES, GO TO QCOM2
17. JSP Q.RANDOM  ^ TYPE RANDOM COMMENT (SECTION B)
18. TCALL 3. [ASCIE] /

19. TRY AGAIN:

20. JST QRP1  1GO TO QRP1

21. QCOM2  MOVE A,TAG  GET ANSWER TAG
22. CAIE A."*"  IS TAG A.**?
23. JST QRP1  1NO, GO TO QRP1

24. TCALL 3. [ASCIE] /

25. NO SUCH CHOICE! TRY AGAIN.

26. QRP1  MOVE A,SVPOS  GET DISK BLOCK CONTAINING TOP OF FRAME
27. CAME A.DSKBLK  SAME AS CURRENT DISK BLOCK
28. USET 1, #SVPOS  IND, GO USE1
29. CAME A.DSKBLK  SAME AS CURRENT DISK BLOCK
30. INPUT 1, IND, INPUT NEW BLOCK
31. MOVER A.DSKBLK  SAVE NEW BLOCK NO.
ACTION MACRO 47(1284)7 00112 2-AUG-74 PAGE 1-10

ACTION MACRO 17-HAY-74

SET POINTER TO BEGINNING OF FRAME

MOVE A,SPVOS-2

PUT BYTE COUNT IN AC-15

INCREMENT DATA INDEX

EXPAND CORE IF NECESSARY

ADD ONE TO FRAME COUNT

GET CURRENT FRAME NO.

TRUE FLG,F-200 I.IS RECOMMEND IN PROGRESS

YES, ADD 1000 TO FRAME NO.

ADD FRAME NO. TO STUD, PERFT DATA

GET TIME OF DAY

CONVERT TO SECONDS

ADD TIME TO STUD, PERFT DATA

GET CURRENT FRAME TYPE

ADD FRAME TYPE TO STUD, PERFT DATA

CLEAR THE WORD

INTERPRET SLIDE STATEMENT

GO TO UNCAC1 IN INTERP, MAC

INTERPRET SLIDE STATEMENT

GO TO GETACT IN INTERP, MAC

INPUT CHARACTER

YES, GO TO EOL IN INTERP, MAC

ETYPE TEXT

TYPE TEXT

CALL 3, CRLFD

JSP Q, C, F, F-1

JMP A, 3, 3

TTCALL 1, A

CAIN A, 15

END OF LINE?

EOL

YES, GO TO EOL IN INTERP, MAC

SECTION I. INTERPRETS AND EXECUTES LOGOUT (L1) STATEMENTS.

SECTION 2. INTERPRETS AND EXECUTES SLIDE (S1) STATEMENTS.
TITLE CPROS

COMMENT %

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 14-MAY-74

PART OF LEARN.MSHTR, VERSION 3

CPROS.MAC PROCESSES CALC MODE STATEMENTS. THE PROCESSED
STATEMENT IS PLACED IN THE CALC STATEMENT BUFFER SO THAT IT
CAN BE EXECUTED BY CALC.MAC. MUCH OF THE CODE IN CPROS.MAC
IS SIMILAR TO THE CODE IN CDRAM.MAC.

ENTRY POINT TO CPROS.MAC

CPROS = FROM CALC.MAC (SECTION A), LOCATED IN SECTION A

EXTERNAL SUBROUTINES CALLED BY CPROS.MAC

EXPCOR = EXPANDS THE LOW SEGMENT IF NECESSARY, LOCATED IN
AUX1.MAC (SECTION H)

FLOUT = "OUTPUTS" THE FLOATING POINT NO. IN AC-A INTO THE
CALC STATEMENT BUFFER

IN == INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B,
LOCATED IN AUX1.MAC (SECTION E)

IOCD = "OUTPUTS" THE INTEGER IN AC-A INTO THE CALC STATEMENT
BUFFER, LOCATED IN AUX1.MAC (SECTION D)

TTYIN = INPUT A FLOATING POINT NO. FROM A USER'S TERMINAL INTO
AC-B, LOCATED IN AUX1.MAC (SECTION F)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOW1.MAC)

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

BUFFNAT = BYTE POINTER TO CALC STATEMENT BUFFER

CLCSTR = CALC STATEMENT BUFFER (16 WORDS)

CONSTR13 (#OSTACK) = OPERATOR STACK (34 WORDS)

CONSTR59 (#HOLD) = USED TO HOLD OPERATOR PRECEDING IN ARRAY
VARIABLE

54
55
56
57
58
59
60
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

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

IDX1 = ADDRESS OF 1ST INDEX VARIABLE

IDXV = FINAL VALUE OF 1ST INDEX VARIABLE

IDX2 = ADDRESS OF 2ND INDEX VARIABLE

IDX2V = INITIAL VALUE OF 2ND INDEX AND FINAL VALUE OF 2ND INDEX
(2 WORDS)

PC4 = SUBROUTINES STORE RETURN ADDRESS HERE

PC5 = SUBROUTINES STORE RETURN ADDRESS HERE

POLST (#STACK) = 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
VARIABLE STORAGE

STYPE = STRING TYPE STACK (72 WORDS)

SYMIDX = SYMBOL TABLE INDEX

SYNTAB = 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

ENTRY CPROS

EXTERN IIC1, IIC2, FLOUT, IODC, TTYIN
EXTERN SYMIDX, STYPE, PSTACK, PTYPE, POLST, CONST
EXTERN PS4, PS5, DATA, IDX1, IDX2, IDXV, IDX2V, EXPGR
EXTERN CLOSTR, ADDR, RSVNUM, SYNTAB, SBUFNT
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>000000</td>
<td>FLGNS=0</td>
</tr>
<tr>
<td>108</td>
<td>000021</td>
<td>A+1</td>
</tr>
<tr>
<td>109</td>
<td>000002</td>
<td>B+2</td>
</tr>
<tr>
<td>110</td>
<td>000003</td>
<td>C+3</td>
</tr>
<tr>
<td>111</td>
<td>000004</td>
<td>D+4</td>
</tr>
<tr>
<td>112</td>
<td>000005</td>
<td>E+5</td>
</tr>
<tr>
<td>113</td>
<td>000006</td>
<td>F+6</td>
</tr>
<tr>
<td>114</td>
<td>000007</td>
<td>G+7</td>
</tr>
<tr>
<td>115</td>
<td>000010</td>
<td>H+10</td>
</tr>
<tr>
<td>116</td>
<td>000016</td>
<td>CFLAGS=16</td>
</tr>
<tr>
<td>117</td>
<td>000017</td>
<td>Q=17</td>
</tr>
<tr>
<td>118</td>
<td>000020</td>
<td>SSTACK=POLLST</td>
</tr>
<tr>
<td>119</td>
<td>000021</td>
<td>OSSSTACK=CONST+13</td>
</tr>
<tr>
<td>120</td>
<td>000022</td>
<td>HOLD=CONST+55</td>
</tr>
<tr>
<td>121</td>
<td>000025</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td></td>
<td></td>
</tr>
<tr>
<td>123</td>
<td></td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>000006</td>
<td>44 07 0 0 000007</td>
</tr>
<tr>
<td>125</td>
<td>000011</td>
<td>44 07 0 0 000006</td>
</tr>
<tr>
<td>126</td>
<td>000002</td>
<td>44 07 0 0 000008</td>
</tr>
<tr>
<td>127</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
<td>SECTION A. PROCESSES ITERATION FIELDS</td>
</tr>
<tr>
<td>129</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>131</td>
<td>000031</td>
<td>620 16 0 0 000007</td>
</tr>
<tr>
<td>132</td>
<td>000004</td>
<td>202 17 0 0 000006</td>
</tr>
<tr>
<td>133</td>
<td>000005</td>
<td>254 08 0 0 000007</td>
</tr>
<tr>
<td>134</td>
<td>000006</td>
<td>851 04 0 0 000001</td>
</tr>
<tr>
<td>135</td>
<td>000010</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>136</td>
<td>000011</td>
<td>254 00 0 0 000006</td>
</tr>
<tr>
<td>137</td>
<td>000012</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>138</td>
<td>000013</td>
<td>254 00 0 0 000006</td>
</tr>
<tr>
<td>139</td>
<td>000014</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>140</td>
<td>000015</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>141</td>
<td>000016</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>142</td>
<td>000017</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>143</td>
<td>000018</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>144</td>
<td>000019</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>145</td>
<td>000020</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>146</td>
<td>000021</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>147</td>
<td>000022</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>148</td>
<td>000023</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>149</td>
<td>000024</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>150</td>
<td>000025</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>151</td>
<td>000026</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>152</td>
<td>000027</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>153</td>
<td>000028</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>154</td>
<td>000029</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>155</td>
<td>000030</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>156</td>
<td>000031</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>157</td>
<td>000032</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>158</td>
<td>000033</td>
<td>368 01 0 0 000000</td>
</tr>
<tr>
<td>159</td>
<td>000034</td>
<td>368 01 0 0 000000</td>
</tr>
</tbody>
</table>

Additional notes: Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION D. EXECUTES ARRAY STATEMENTS.

ARRAY I TDCALL 4, A INPUT CHARACTER

CAI B A 11

CAI B A 48

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL "A"

CALL "H"

JST ARERR IF CHARACTER IS NOT A LETTER, GO TYPE ERROR MESSAGE

SET2 G

MOVE C,POINT1 GET BYTE POINTER TO AC-G

IDPS A,C INPUT CHARACTER IN AC-G

TDCALL 4, A INPUT CHARACTER

CALL "M"

CALL "H"

JST ARERR IF CHARACTER IS NOT A LETTER, GO TYPE ERROR MESSAGE

JST ARERR INPUT CHARACTER IN AC-G

TDCALL 4, A INPUT NEXT CHARACTER

CALL "A" "H" IF CHARACTER A "A" OR

JST ARERR IF CHARACTER A "A"

JST BCRERR NO, GO TYPE ERROR MESSAGE (SECT. I)

TDCALL 4, A INPUT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

JSP 0 1102 IF INPUT NO. OF ROWS IN ARRAY

JST 1 2

TDCALL 4, A INPUT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

JSP 0 1102 IF INPUT NO. OF COLUMNS IN ARRAY

JST 1 2 INPUT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

JSP 0 1102 IF INPUT NO. OF COLUMNS IN ARRAY

JST 1 2 INPUT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11

JST 1 3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

CALL 11
I

JRST BERR

INQ, GO TYPE ERROR MESSAGE (SECTION I)

IMUL E,B

ICOMPUTE NUMBER OF WORDS IN THE ARRAY

ARRY1

AOS D SYMTAB I GET INDEX TO END OF SYMBOL TABLE

CPS3371

350 04 0 0 0

CPS3341

350 04 0 0 0

376

003343

173 00 0 0 0

JRST TABERR

YES, GO TYPE ERROR MESSAGE (SECTION I)

JRST POCS4

MORE THAN 6000 WORDS REQUESTED?

MOVC F,B

ELSE, PUT NO. OF ROWS IN AC-E

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?

CALL A,10210

MORE THAN 6000 WORDS REQUESTED?
I SECTION G. PROCESSES ATYPE STATEMENTS.

671 00795 1 201 01 00 006765  ATYPE1 MOVE 1,5 INPUT ATYPE STATEMENT CODE IN AC-A
672 00796 1 136 01 00 00752  ATYPE1 MOVE 1,5 INPUT ATYPE STATEMENT CODE IN AC-A
673 00797 1 051 04 00 00001  IDPB A, BUFNT INPUT LINE FEED
674 00798 1 302 01 00 00011  IDPB A, BUFNT INPUT CHARACTER
675 00799 1 366 01 00 00204  CAIN A,101
676 00800 1 254 00 00 00756  J R S T - 1,3  IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
677 00801 1 301 01 00 00701  CALL A, "A"
678 00802 1 303 01 00 00132  CALL A, "2"
679 00803 1 254 01 00 00373  J R S T ARERR IF CHARACTER IS NOT A LETTER, TYPE ERROR MESSAGE (SECT. D)
680 00804 1 483 37 00 00300  SETZ G
681 00805 1 203 03 00 00000  MOVE G,POINTER TO AC-G
682 00806 1 136 01 00 00233  IDPB A,C INPUT CHARACTER IN AC-G
683 00807 1 251 24 00 00301  CALL A, "A"
684 00808 1 301 01 00 00132  CALL A, "2"
685 00809 1 254 02 00 00373  J R S T ARERR IF CHARACTER IS NOT A LETTER, TYPE ERROR MESSAGE (SECT. D)
686 00810 1 136 01 00 00383  IDPB A,C INPUT CHARACTER IN AC-G
Icosa
<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>796</td>
<td>001151</td>
<td>MOVEM D:STYPE IN STRING TYPE STACK</td>
</tr>
<tr>
<td>797</td>
<td>001121</td>
<td>MOVEI G,1 SET STACK INDEX TO 1</td>
</tr>
<tr>
<td>798</td>
<td>001131</td>
<td>CALL C:&quot;A&quot;</td>
</tr>
<tr>
<td>799</td>
<td>001151</td>
<td>CAILE C:&quot;E&quot; IIF NOT ARRAY VARIABLE; JUMP</td>
</tr>
<tr>
<td>800</td>
<td>001151</td>
<td>JST,+3 IIF ARRAY VARIABLE; JUMP</td>
</tr>
<tr>
<td>801</td>
<td>001151</td>
<td>254 00 00 00 00 001646</td>
</tr>
<tr>
<td>802</td>
<td>001151</td>
<td>254 00 00 00 00 001647</td>
</tr>
<tr>
<td>803</td>
<td>001151</td>
<td>254 00 00 00 00 001648</td>
</tr>
<tr>
<td>804</td>
<td>001151</td>
<td>254 00 00 00 00 001649</td>
</tr>
<tr>
<td>805</td>
<td>001151</td>
<td>254 00 00 00 00 001650</td>
</tr>
<tr>
<td>806</td>
<td>001151</td>
<td>254 00 00 00 00 001651</td>
</tr>
<tr>
<td>807</td>
<td>001151</td>
<td>254 00 00 00 00 001652</td>
</tr>
<tr>
<td>808</td>
<td>001151</td>
<td>254 00 00 00 00 001653</td>
</tr>
<tr>
<td>809</td>
<td>001151</td>
<td>254 00 00 00 00 001654</td>
</tr>
<tr>
<td>810</td>
<td>001151</td>
<td>254 00 00 00 00 001655</td>
</tr>
<tr>
<td>811</td>
<td>001151</td>
<td>254 00 00 00 00 001656</td>
</tr>
</tbody>
</table>

**SECTION 1. POLISH NOTATION GENERATOR.**

### POLLY1
- MOVEM Q,PC5 ISAVE RETURN ADDRESS
- JST RVAR1 IGO TO RVAR1
- MOVEM Q,PC5 ISAVE RETURN ADDRESS
- JST AVAR1 IGO TO AVAR1
- MOVEM Q,PC5 ISAVE RETURN ADDRESS
- JST POL1 IGO TO POL1
- TTALL 4,A INPUT CHARACTER

### POLLI1
- MOVEM A,48
- JST +3 IIF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
- MOVEM A,49 IIS CHARACTER A NUMBER PLUS?
- MOVEM A,46 IIS CHARACTER A MINUS?
- MOVEM A,47 IIS CHARACTER A MINUS?
- MOVEM A,48 IIS CHARACTER A MINUS?
- MOVEM A,45 IIS CHARACTER A MINUS?
- MOVEM A,44 IIS CHARACTER A MINUS?
- MOVEM A,43 IIS CHARACTER A MINUS?
- MOVEM A,42 IIS CHARACTER A MINUS?
- MOVEM A,41 IIS CHARACTER A MINUS?
- MOVEM A,40 IIS CHARACTER A MINUS?
- MOVEM A,39 IIS CHARACTER A MINUS?
- MOVEM A,38 IIS CHARACTER A MINUS?
- MOVEM A,37 IIS CHARACTER A MINUS?
- MOVEM A,36 IIS CHARACTER A MINUS?
- MOVEM A,35 IIS CHARACTER A MINUS?
- MOVEM A,34 IIS CHARACTER A MINUS?
- MOVEM A,33 IIS CHARACTER A MINUS?
- MOVEM A,32 IIS CHARACTER A MINUS?
- MOVEM A,31 IIS CHARACTER A MINUS?
- MOVEM A,30 IIS CHARACTER A MINUS?
- MOVEM A,29 IIS CHARACTER A MINUS?
- MOVEM A,28 IIS CHARACTER A MINUS?
- MOVEM A,27 IIS CHARACTER A MINUS?
- MOVEM A,26 IIS CHARACTER A MINUS?
- MOVEM A,25 IIS CHARACTER A MINUS?
- MOVEM A,24 IIS CHARACTER A MINUS?
- MOVEM A,23 IIS CHARACTER A MINUS?
- MOVEM A,22 IIS CHARACTER A MINUS?
- MOVEM A,21 IIS CHARACTER A MINUS?
- MOVEM A,20 IIS CHARACTER A MINUS?
- MOVEM A,19 IIS CHARACTER A MINUS?
- MOVEM A,18 IIS CHARACTER A MINUS?
- MOVEM A,17 IIS CHARACTER A MINUS?
- MOVEM A,16 IIS CHARACTER A MINUS?
- MOVEM A,15 IIS CHARACTER A MINUS?
- MOVEM A,14 IIS CHARACTER A MINUS?
- MOVEM A,13 IIS CHARACTER A MINUS?
- MOVEM A,12 IIS CHARACTER A MINUS?
- MOVEM A,11 IIS CHARACTER A MINUS?
- MOVEM A,10 IIS CHARACTER A MINUS?
- MOVEM A,9 IIS CHARACTER A MINUS?
- MOVEM A,8 IIS CHARACTER A MINUS?
- MOVEM A,7 IIS CHARACTER A MINUS?
- MOVEM A,6 IIS CHARACTER A MINUS?
- MOVEM A,5 IIS CHARACTER A MINUS?
- MOVEM A,4 IIS CHARACTER A MINUS?
- MOVEM A,3 IIS CHARACTER A MINUS?
- MOVEM A,2 IIS CHARACTER A MINUS?
- MOVEM A,1 IIS CHARACTER A MINUS?
SECTION J. PLACES VARIABLE IN SYMBOL TABLE AND OUTPUTS ITS ADDRESS.

ASVARI: MOVE H.A SAVE THE CHARACTER IN AC-A
MOVE D.SYMIDX GET POINTER TO END OF SYMBOL TABLE
ASVII: HHLE A.SYMID (D) GET SYMBOL FROM SYMBOL TABLE
CANNY A.C IS VARIABLE SAME AS SYMBOL?
JSTL +3 YES, JUMP
JSTL +3 NO SYM IF NAME NOT FOUND, GO TO NOSYM
MOVE A.2 PUT VARIABLE CODE <02> IN AC-A
IDPB A.SYMID OUTPUT <02>
HRRZ A.SYMID (D) GET ADDRESS OF VARIABLE
IDIV A.1098
ADDI A.40
ADDI B.40
NO ERRORS DETECTED

PROGRAM BREAK IS 002655

4K CORE USED
<table>
<thead>
<tr>
<th>ATYPE</th>
<th>AVAR</th>
<th>AVER1</th>
<th>AVER2</th>
<th>AVER3</th>
</tr>
</thead>
<tbody>
<tr>
<td>290</td>
<td>673#</td>
<td>982#</td>
<td>819</td>
<td>939</td>
</tr>
<tr>
<td>122</td>
<td>123</td>
<td>123</td>
<td>122</td>
<td>122</td>
</tr>
<tr>
<td>118#</td>
<td>118#</td>
<td>118#</td>
<td>118#</td>
<td>118#</td>
</tr>
<tr>
<td>116#</td>
<td>116#</td>
<td>116#</td>
<td>116#</td>
<td>116#</td>
</tr>
<tr>
<td>112#</td>
<td>112#</td>
<td>112#</td>
<td>112#</td>
<td>112#</td>
</tr>
<tr>
<td>108#</td>
<td>108#</td>
<td>108#</td>
<td>108#</td>
<td>108#</td>
</tr>
<tr>
<td>106#</td>
<td>106#</td>
<td>106#</td>
<td>106#</td>
<td>106#</td>
</tr>
<tr>
<td>104#</td>
<td>104#</td>
<td>104#</td>
<td>104#</td>
<td>104#</td>
</tr>
<tr>
<td>100#</td>
<td>100#</td>
<td>100#</td>
<td>100#</td>
<td>100#</td>
</tr>
<tr>
<td>98#</td>
<td>98#</td>
<td>98#</td>
<td>98#</td>
<td>98#</td>
</tr>
<tr>
<td>96#</td>
<td>96#</td>
<td>96#</td>
<td>96#</td>
<td>96#</td>
</tr>
<tr>
<td>94#</td>
<td>94#</td>
<td>94#</td>
<td>94#</td>
<td>94#</td>
</tr>
<tr>
<td>92#</td>
<td>92#</td>
<td>92#</td>
<td>92#</td>
<td>92#</td>
</tr>
<tr>
<td>90#</td>
<td>90#</td>
<td>90#</td>
<td>90#</td>
<td>90#</td>
</tr>
<tr>
<td>88#</td>
<td>88#</td>
<td>88#</td>
<td>88#</td>
<td>88#</td>
</tr>
<tr>
<td>86#</td>
<td>86#</td>
<td>86#</td>
<td>86#</td>
<td>86#</td>
</tr>
<tr>
<td>84#</td>
<td>84#</td>
<td>84#</td>
<td>84#</td>
<td>84#</td>
</tr>
<tr>
<td>82#</td>
<td>82#</td>
<td>82#</td>
<td>82#</td>
<td>82#</td>
</tr>
<tr>
<td>80#</td>
<td>80#</td>
<td>80#</td>
<td>80#</td>
<td>80#</td>
</tr>
<tr>
<td>78#</td>
<td>78#</td>
<td>78#</td>
<td>78#</td>
<td>78#</td>
</tr>
<tr>
<td>76#</td>
<td>76#</td>
<td>76#</td>
<td>76#</td>
<td>76#</td>
</tr>
<tr>
<td>74#</td>
<td>74#</td>
<td>74#</td>
<td>74#</td>
<td>74#</td>
</tr>
<tr>
<td>72#</td>
<td>72#</td>
<td>72#</td>
<td>72#</td>
<td>72#</td>
</tr>
<tr>
<td>70#</td>
<td>70#</td>
<td>70#</td>
<td>70#</td>
<td>70#</td>
</tr>
<tr>
<td>68#</td>
<td>68#</td>
<td>68#</td>
<td>68#</td>
<td>68#</td>
</tr>
<tr>
<td>66#</td>
<td>66#</td>
<td>66#</td>
<td>66#</td>
<td>66#</td>
</tr>
<tr>
<td>64#</td>
<td>64#</td>
<td>64#</td>
<td>64#</td>
<td>64#</td>
</tr>
<tr>
<td>62#</td>
<td>62#</td>
<td>62#</td>
<td>62#</td>
<td>62#</td>
</tr>
<tr>
<td>60#</td>
<td>60#</td>
<td>60#</td>
<td>60#</td>
<td>60#</td>
</tr>
<tr>
<td>58#</td>
<td>58#</td>
<td>58#</td>
<td>58#</td>
<td>58#</td>
</tr>
<tr>
<td>56#</td>
<td>56#</td>
<td>56#</td>
<td>56#</td>
<td>56#</td>
</tr>
<tr>
<td>54#</td>
<td>54#</td>
<td>54#</td>
<td>54#</td>
<td>54#</td>
</tr>
<tr>
<td>52#</td>
<td>52#</td>
<td>52#</td>
<td>52#</td>
<td>52#</td>
</tr>
<tr>
<td>50#</td>
<td>50#</td>
<td>50#</td>
<td>50#</td>
<td>50#</td>
</tr>
<tr>
<td>48#</td>
<td>48#</td>
<td>48#</td>
<td>48#</td>
<td>48#</td>
</tr>
<tr>
<td>46#</td>
<td>46#</td>
<td>46#</td>
<td>46#</td>
<td>46#</td>
</tr>
<tr>
<td>44#</td>
<td>44#</td>
<td>44#</td>
<td>44#</td>
<td>44#</td>
</tr>
<tr>
<td>42#</td>
<td>42#</td>
<td>42#</td>
<td>42#</td>
<td>42#</td>
</tr>
<tr>
<td>40#</td>
<td>40#</td>
<td>40#</td>
<td>40#</td>
<td>40#</td>
</tr>
<tr>
<td>38#</td>
<td>38#</td>
<td>38#</td>
<td>38#</td>
<td>38#</td>
</tr>
<tr>
<td>36#</td>
<td>36#</td>
<td>36#</td>
<td>36#</td>
<td>36#</td>
</tr>
<tr>
<td>34#</td>
<td>34#</td>
<td>34#</td>
<td>34#</td>
<td>34#</td>
</tr>
<tr>
<td>32#</td>
<td>32#</td>
<td>32#</td>
<td>32#</td>
<td>32#</td>
</tr>
<tr>
<td>30#</td>
<td>30#</td>
<td>30#</td>
<td>30#</td>
<td>30#</td>
</tr>
<tr>
<td>28#</td>
<td>28#</td>
<td>28#</td>
<td>28#</td>
<td>28#</td>
</tr>
<tr>
<td>26#</td>
<td>26#</td>
<td>26#</td>
<td>26#</td>
<td>26#</td>
</tr>
<tr>
<td>24#</td>
<td>24#</td>
<td>24#</td>
<td>24#</td>
<td>24#</td>
</tr>
<tr>
<td>22#</td>
<td>22#</td>
<td>22#</td>
<td>22#</td>
<td>22#</td>
</tr>
<tr>
<td>20#</td>
<td>20#</td>
<td>20#</td>
<td>20#</td>
<td>20#</td>
</tr>
<tr>
<td>18#</td>
<td>18#</td>
<td>18#</td>
<td>18#</td>
<td>18#</td>
</tr>
<tr>
<td>16#</td>
<td>16#</td>
<td>16#</td>
<td>16#</td>
<td>16#</td>
</tr>
<tr>
<td>14#</td>
<td>14#</td>
<td>14#</td>
<td>14#</td>
<td>14#</td>
</tr>
<tr>
<td>12#</td>
<td>12#</td>
<td>12#</td>
<td>12#</td>
<td>12#</td>
</tr>
<tr>
<td>10#</td>
<td>10#</td>
<td>10#</td>
<td>10#</td>
<td>10#</td>
</tr>
<tr>
<td>8#</td>
<td>8#</td>
<td>8#</td>
<td>8#</td>
<td>8#</td>
</tr>
<tr>
<td>6#</td>
<td>6#</td>
<td>6#</td>
<td>6#</td>
<td>6#</td>
</tr>
<tr>
<td>4#</td>
<td>4#</td>
<td>4#</td>
<td>4#</td>
<td>4#</td>
</tr>
<tr>
<td>2#</td>
<td>2#</td>
<td>2#</td>
<td>2#</td>
<td>2#</td>
</tr>
<tr>
<td>1#</td>
<td>1#</td>
<td>1#</td>
<td>1#</td>
<td>1#</td>
</tr>
<tr>
<td>0#</td>
<td>0#</td>
<td>0#</td>
<td>0#</td>
<td>0#</td>
</tr>
<tr>
<td>FUNCS</td>
<td>MACRO 47(204)-7 00117 2-AUG-74 PAGE 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TITLE</td>
<td>FUNCS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMMENT</td>
<td>K</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PREPARED</td>
<td>BY GARRET A. VANDER LUGT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPLETED</td>
<td>14-MAY-74</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PART OF</td>
<td>LEARN, SHR, VERSION 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THE MATHEMATICAL FUNCTIONS CONTAINED IN FUNCS.MAC ARE SLIGHTLY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODIFIED VERSIONS OF THE FORTRAN SCIENCE LIBRARY FUNCTIONS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUPPLIED BY DIGITAL EQUIPMENT CORPORATION (1959).</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CONTENTS OF FUNCS.MAC**

<table>
<thead>
<tr>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECTION A</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>SECTION B</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>SECTION C</td>
</tr>
<tr>
<td>23</td>
</tr>
<tr>
<td>SECTION D</td>
</tr>
<tr>
<td>25</td>
</tr>
<tr>
<td>SECTION E</td>
</tr>
<tr>
<td>27</td>
</tr>
<tr>
<td>SECTION F</td>
</tr>
<tr>
<td>29</td>
</tr>
<tr>
<td>SECTION G</td>
</tr>
<tr>
<td>31</td>
</tr>
<tr>
<td>SECTION H</td>
</tr>
<tr>
<td>33</td>
</tr>
<tr>
<td>SECTION I</td>
</tr>
<tr>
<td>35</td>
</tr>
<tr>
<td>SECTION J</td>
</tr>
<tr>
<td>37</td>
</tr>
<tr>
<td>SECTION K</td>
</tr>
<tr>
<td>39</td>
</tr>
<tr>
<td>SECTION L</td>
</tr>
<tr>
<td>40</td>
</tr>
<tr>
<td>SECTION M</td>
</tr>
<tr>
<td>41</td>
</tr>
<tr>
<td>SECTION N</td>
</tr>
<tr>
<td>42</td>
</tr>
<tr>
<td>SECTION O</td>
</tr>
<tr>
<td>43</td>
</tr>
<tr>
<td>SECTION P</td>
</tr>
<tr>
<td>44</td>
</tr>
<tr>
<td>SECTION Q</td>
</tr>
</tbody>
</table>

**ENTRY POINTS TO FUNCS.MAC**
<table>
<thead>
<tr>
<th>FUNCS</th>
<th>MACRO</th>
<th>47(204)</th>
<th>7-88</th>
<th>I7</th>
<th>2-AUG-74</th>
<th>PAGE 1-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>FABS</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>FACS</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>FACOS</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION P</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>FACOSD</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>FASIN</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>FASIND</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>FATAN</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>FATAND</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>FCOS</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION J</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>FCOSD</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>FCOS</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION E</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>FINT</td>
<td>FROM INTERP.MAC (SECTION F) AND CALC.HAC (SECTIONS F &amp; O)</td>
<td>LOCATED IN SECTION B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>FLOG</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>FNLOG</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>FRAN</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>FSIN</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>FSIND</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>FSQRT</td>
<td>FROM CALC.HAC (SECTION O)</td>
<td>LOCATED IN SECTION D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td></td>
<td>EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOHM1.MAC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td></td>
<td>FIELD = USED TO STORE INTERMEDIATE RESULTS OF CALCULATIONS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td></td>
<td>RANNUM = INTEGER RANDOM NO, USED BY RANDOM FUNCTION</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
<td>DESCRIPTION OF CFLAGS STORED IN AC-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td></td>
<td>BIT 31 = NEGATIVE ARGUMENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td></td>
<td>000001 = WISEG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td></td>
<td>ENTRY FATAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECTION A. ABSOLUTE VALUE SUBROUTINE:

000000 217 00 00 000001 FABSI MOVMS A
000001 254 00 17 000000 JST (C)

SECTION B. INTEGER SUBROUTINE:

000022 620 16 00 000023 FINI TRZ CFLAGS, 20
000031 322 01 00 000030 JUMPE A, (G)
000034 327 01 00 000037 JUMPC A, +3
000054 217 00 00 000001 MOVMS A
000066 660 16 00 000020 TRD CFLAGS, 20
000077 200 03 00 000030 MOVE C, A
000081 242 03 00 777748 LSH C, +3
0000a1 620 01 00 777700 TLZ A, 777700
000121 275 03 00 000023 SUBI C, +33
000213 242 01 00 000000 LSH A, (C)
000214 600 16 00 000020 TRNE CFLAGS, 20
000019 217 00 00 000001 MOVMS A
00001a 254 00 17 000000 JST (C)

SECTION C. RANDOM NUMBER GENERATOR:

000021 200 01 00 000455 FRAN: MOVE A, [R6360016]
000020 224 01 00 000000 MUL A, RANNUM
000021 244 01 00 000004 ASHC A, 4
000022 242 02 00 777748 LSH B, -4
000023 270 01 00 000002 ADD A, B
000024 623 01 00 760000 TLEA A, 760000
000025 271 01 00 000001 ADDI A, 1
SECTION D. SQUARE ROOT SUBROUTINE.

```
160 000261 202 01 00 000021  MOVEM A,RANUM
161 000269 554 02 00 000081  MLRE B,A
162 000266 132 02 00 000216  FSC B,16
163 000261 505 01 00 000100  HRL1 A,0
164 000262 132 01 00 000174  FSC A,174
165 000263 140 01 00 000202  FAD A,B
166 000264 254 20 0 17 000000  JRST (Q)

ISECTION E. EXPONENTIAL SUBROUTINE.
```

```
000073 200 02 00 000001  FEXP1  MOVE B,A
000074 317 02 00 000154  CAMG B,E7
000075 254 00 00 000183  JRST OUT2
000076 317 02 00 000153  CAMG B,E7
```
**FUNCTION MACRO 49(204)-7 08117 2-AUG-74 PAGE I-5**

**SECTION F. COMMON LOGARITHM SUBROUTINE.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>266</td>
<td>MOVHS A</td>
<td></td>
</tr>
<tr>
<td>267</td>
<td>JUMPE A,ZERO</td>
<td></td>
</tr>
<tr>
<td>268</td>
<td>MOVS A</td>
<td></td>
</tr>
<tr>
<td>269</td>
<td>JUMPE A,ZERO</td>
<td></td>
</tr>
<tr>
<td>270</td>
<td>MOVE F,Q</td>
<td></td>
</tr>
<tr>
<td>271</td>
<td>JSP G,FLNC+2</td>
<td></td>
</tr>
<tr>
<td>272</td>
<td>FPQ A,LOG10A</td>
<td></td>
</tr>
<tr>
<td>273</td>
<td>JRST (F)</td>
<td></td>
</tr>
<tr>
<td>274</td>
<td>LOG10A,77747737305</td>
<td></td>
</tr>
</tbody>
</table>

**SECTION G. NATURAL LOGARITHM SUBROUTINE.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>280</td>
<td>MOVHS A</td>
<td></td>
</tr>
<tr>
<td>281</td>
<td>JUMPE A,ONE</td>
<td></td>
</tr>
<tr>
<td>282</td>
<td>CMMN A,ONE</td>
<td></td>
</tr>
<tr>
<td>283</td>
<td>JRST ZERAN</td>
<td></td>
</tr>
<tr>
<td>284</td>
<td>ASHC A,-33</td>
<td></td>
</tr>
<tr>
<td>285</td>
<td>ADD A,210000</td>
<td></td>
</tr>
<tr>
<td>286</td>
<td>MOVSH A,C</td>
<td></td>
</tr>
<tr>
<td>287</td>
<td>MOVS A,567377</td>
<td></td>
</tr>
<tr>
<td>288</td>
<td>FAD A,C</td>
<td></td>
</tr>
<tr>
<td>289</td>
<td>ASHC B,13</td>
<td></td>
</tr>
<tr>
<td>290</td>
<td>TIC B,200000</td>
<td></td>
</tr>
<tr>
<td>291</td>
<td>FAD B,L</td>
<td></td>
</tr>
<tr>
<td>292</td>
<td>MOVE A,B</td>
<td></td>
</tr>
<tr>
<td>293</td>
<td>FAD A,L2</td>
<td></td>
</tr>
<tr>
<td>294</td>
<td>FDV B,A</td>
<td></td>
</tr>
<tr>
<td>295</td>
<td>MOVE B,E</td>
<td></td>
</tr>
<tr>
<td>296</td>
<td>MOVS A,E</td>
<td></td>
</tr>
<tr>
<td>297</td>
<td>MOVHE B,E</td>
<td></td>
</tr>
<tr>
<td>298</td>
<td>MOVS A,B</td>
<td></td>
</tr>
<tr>
<td>299</td>
<td>FMP B,A</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>MOV A,L3</td>
<td></td>
</tr>
<tr>
<td>301</td>
<td>MOVS A,L3</td>
<td></td>
</tr>
<tr>
<td>302</td>
<td>FMP A,B</td>
<td></td>
</tr>
<tr>
<td>303</td>
<td>FAD A,L4</td>
<td></td>
</tr>
<tr>
<td>304</td>
<td>FAD A,E</td>
<td></td>
</tr>
<tr>
<td>305</td>
<td>FMP A,L7</td>
<td></td>
</tr>
<tr>
<td>306</td>
<td>JRST (G)</td>
<td></td>
</tr>
<tr>
<td>307</td>
<td>ZER01 TTCCAL 3,1ASC12</td>
<td></td>
</tr>
<tr>
<td>308</td>
<td>ATTEMPTED LOG OF ZERO</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- MOV A,MINF
- JRST (G)
- ZERAN1 SETE A,
- JRST (G)
- ONE 201400000000
- L1 578257514542
- L2 201952023352
- L3 20842532521
- L4 200754213684
- L5 222561251982
TITLE AUX1

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT  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 & 0), 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 I), LOCATED IN SECTION G

GETBUF = FROM INTERP.MAC (SECTION A), LOCATED IN SECTION B

IIC = FROM CPROS.MAC (SECTIONS F & G), LOCATED IN SECTION E

IIC2 = FROM CPROS.MAC (SECTIONS A, C, D, E, F & G), LOCATED
IN SECTION E

IIO = FROM INTERP.MAC (SECTIONS B, EE & GG) AND ACTION.MAC
(SECTION K), LOCATED IN SECTION E

INFLP - FROM INTERP.MAC (SECTION J) AND CALC.MAC (SECTIONS L & G), LOCATED IN SECTION F

10DC - FROM CPROS.MAC (SECTIONS E, F & G), LOCATED IN SECTION D

IOT - FROM INTERP.MAC (SECTION B), LOCATED IN SECTION D

OUTCHR - FROM INTERP.MAC (SECTIONS BB & HH), LOCATED IN SECTION C

OUTFLP - FROM CALC.MAC (SECTION G & H) AND ACH2.MAC (SECTION E), LOCATED IN SECTION C

TTYIN - FROM CALC.MAC (SECTION F) AND CPROS.MAC (SECTION H), LOCATED IN SECTION F

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LOW1.MAC)

BUFFNT - BYTE POINTER TO CALC STATEMENT BUFFER

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

FIELD - TOTAL FIELD WIDTH AND NO. OF DIGITS AFTER DECIMAL POINT (2 WORDS)

IBUF - INPUT BUFFER HEADER (3 WORDS)

IO - INPUT OR OUTPUT INSTRUCTION

LESRD1 (#EXPNT) - BASE TEN EXPONENT

LESRD2 (#DIGITS) - NO. OF SPACES WHICH MUST BE TYPED AFTER A LEFT-justified NO.

OBUF - OUTPUT BUFFER HEADER (3 WORDS)

PDST - PUSH DOWN LIST FOR STORING ASCII DIGITS OF A NUMBER BEING OUTPUT

STORE (#DSKBLK) - CURRENT DISK BLOCK

DESCRIPTION OF FLAGS STORED IN AO-16

BIT 25 NEGATIVE NO.

%
SECTION A. INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL AN END OF LINE IS ENCOUNTERED.

ENTRY GETBUF

INTERN GETCHR, IOT, ID, INFLP, OUTFLP, TTYIN

EXTERN EDLIN, EXPCOM, FLOUT, IDOC, IIC, I122

EXTERN DATA, ID, BUFPINT

EXERN IBUF, OBUF, POLST, STORE, LESHRO, FIELD

FLAGS=0
A=1
B=2
C=3
D=4
E=5
F=6
INDEX=6
G=7
H=10
I=16
O=17
K110=1; FOR K110 CPU, LET K110=0

SECTION B. FILLS DISK INPUT BUFFER FOR DISK INPUT ROUTINE WHICH IS LOCATED IN FAST MEMORY.

GETBUF: AOS DSKBLK; INCREMENT DISK BLOCK NO.

USEI 1; DSKBLK; TELL THE MONITOR WHICH BLOCK TO GET NEXT

INPUT DISK BLOCK

IF NO ERROR, GO TO RSTACS

END OF FILE?

DISK INPUT ERROR ON CHANNEL 1 = AUX1

HALT
I SECTION C. OUTPUTS A CHARACTER FROM AC-A TO THE DISK
ION CHANNEL 2,

..-

I SECTION D. OUTPUTS AN INTEGER FROM AC-A TO A USER'S TERMINAL
FOR THE CALC STATEMENT BUFFER.

..-

I SECTION E. INPUTS AN INTEGER FROM A USER'S TERMINAL OR THE
DISK INTO AC-B.
SECTION F. INPUTS A NUMBER FROM A USER'S TERMINAL OR FROM THE CALC STATEMENT BUFFER AND CONVERTS IT TO FLOATING POINT. THE CONVERTED NUMBER IS PLACED IN AC-B.
THE ROUTINE BELOW CONVERTS THE NO. IN AC-D TO FLOATING POINT.

IF NEEDED FOR THE K110 CPU.

IF E K110, <FLOAT A> IDIVIDE NO. INTO TWO PARTS

SKE B, 400000 IDIVIDE HIGH ORDER PART IS ZERO

TLC B, 254000 IDIVIDE HIGH ORDER PART TO FLOATING POINT

TLC C, 233000 IDIVIDE LOW ORDER PART TO FLOATING POINT.

FAD 8,C IDIVIDE THE TWO PARTS AND NORMALIZE

JRT (0) RETURN
THE ROUTINE BELOW CONVERTS THE NO. IN AC-B TO 10 DIGIT INTEGER (K10 CPU ONLY)
MOVEI B,1
ELSE SET NO. OF DIGITS BEFORE DECIMAL POINT TO 1
SOSG.C.EXPNT
INPUT EXPONENT MINUS ONE IN AC-6
GETDIGI
POP D,A
GET CHARACTER FROM PUSH DOWN LIST
SKIPGE FIELD
STANDARD FORMAT?
JST A
YES, JUMP
JUMP IF Digit IN FRONT OF DECIMAL POINT
SOSGE FIELD+1
DECIMAL POINT, NO. OF DIGITS AFTER ",",?
JST TYPEXP
JUMP IF ENOUGH DIGITS HAVE BEEN TYPED
SOS DIGITS
DECIMAL DIGITS
XCT IO
OUTPUT DECIMAL POINT
XCT 10
OUTPUT CHARACTER
JUMP E,TYPEXP
IF NO MORE CHARACTERS, GO TO TYPEXP
SOUH B,*+6
JUMP IF PERIOD DOES NOT GO HERE
MOVEI A,","
SOS DIGITS
DECIMAL TOTAL DIGITS
.Skip Field
Skip if standard format
Skip Field+1
Skip if second format no. is zero
XCT IO
Output decimal point
SOJE.A,GETDIGI
Go get next character from push down list
JST A,*+6
YES, JUMP
.Skip Field+1
Skip if digits after ",",?
JST A,*+4
YES, JUMP
MOVEI A,40
Put a space in AC-A
XCT IO
Output a space
.SOS FIELD
Skip another space if necessary
JUMP C,(G
Return if exponent is zero
MOVNT A
Skip Field
Skip if standard format
XCT 10
Output a space
.ADD A,DIGITS
Subtract 4 from total digits
MOVEI A,","
Output ","
JUMP C,POSEX
JUMP IF exponent is negative
MOVEI A,","-
MOVHS C
Take absolute value of exponent.
.JST A,*+2
Input minus sign in AC-A
.POSEX
Input plus sign in AC-A
XCT 10
Input plus or minus
.IDIVI C+12
Separate exponent into two digits
.IORI C,60
Convert to ASCII code
.MOVE A,C
Convert a character
.XCT IO
Convert 2nd digit to ASCII code
.MOVE A,D
.XCT IO
Output character
.XCT IO
Output space in AC-A
.XCT IO
Output a space
.SOS FIELD
Skip field
.Skip Field
Skip if standard format
.XCT IO
Output a space
.SOS DIGITS
Decrement total digits
.XCT IO
Output a zero
.Skip Field
Standard format?
.JST A,0
YES, return
SECTION H. EXPANDS THE LOW SEGMENT IF NEEDED TO STORE 
STUDENT PERFORMANCE DATA.

603 00564' 201 02 0 6 00001* EXCPEND MOVEI B,DATA+1(INDEX). GET HIGHEST LOWSEG ADDRESS TO BE REFERENCED
602 00565' 317 02 0 0 080044 CANG B,JOBCOL. IS THAT ADDRESS WITHIN THE CURRENT LOWSEG?
603 00566' 254 00 0 17 000000 JST (Q). YES, RETURN
604 00567' 047 02 0 0 000011 CORE B. REQUEST MORE CORE
605 00568' 254 00 0 000071 JRST (+2) GO TYPE ERROR MESSAGE IF CORE CAPACITY EXCEEDED
606 00569' 254 00 0 17 000000 JST (Q) RETURN
607 TCALL 3,1ASC1Z /
608 00570' 051 03 0 00 0000634* /CORE_CAPACITY_EXCEEDED
609 00571' 254 04 0 00 000000 HALT
610 00572' 254 00 0 00 000000 NOSTM
611 END

NO ERRORS DETECTED.

PROGRAM BREAK IS 000642

3K CORE USED
NO ERRORS DETECTED

PROGRAM BREAK IS 001554

3K CORE USED
TITLE LEDIT

COMMENT %

WRITTEN BY GARREY 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. PERI.MAC
3. LPRINT.MAC
4. WRITED.MAC
5. AUX2.MAC
6. LDTLOW.MAC (DUMMY LOW SEGMENT)

THESE PROGRAMS SHOULD BE LOADED IN THE ORDER IN WHICH THEY ARE LISTED
ABOVE. AFTER LOADING, A REENTERABLE, BINARY CORE IMAGE OF THE LEDIT
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SSAVE 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.
WRT - EXECUTES A "WRITE" COMMAND, LOCATED IN WRTED,MAC (SECTION A)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LDLOW,MAC)

ARGMNT - ARGUMENTS FOR GETSEG UDO (6 WORDS)

STORE - COMMAND INDEX IS SAVED HERE DURING A GETSEG

TRMBLK -TRMHOP, ARGUMENTS (3 WORDS)

% 000000  MISEG

EXTERN ARGMNT, GPy, EDt, EDT2, EOLINE, Fils, LOST, RENAM, STORE, WRT, TRMBLK

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 LEARN,MAC, LRPRRT,MAC, OR LPROS,MAC.

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

000001  047 01 0 00 0000 GETSEG: GETSEG A,
000001  254 04 0 00 0000 HALT

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.

000002  203 02 0 00 0000 RETURN, MOVE B, STORE "GET THE COMMAND INDEX IN AO-B"
000003  254 00 1 02 00003 JRT 9, (B)
000004  00000 000217 XND @, WRITE; SEE SECTION 1
000004  00000 000213 XND @, EDIT; SEE SECTION 1
000004  00000 000211 XND @, LIST; SEE SECTION 1
000007  00000 000207 XND @, FILES; SEE SECTION 1
000010  00000 000209 XND @, RENAME; SEE SECTION 1
000011  00000 000215 XND @, EDIT2; SEE SECTION 1
SECTION C. INPUT THE FIRST WORD IN A LEARN SYSTEM COMMAND.

160 000012 000009 000014  
161 000013 000000 000021  
162 000009  
163  
164  
165  
166 000014 260 01 0 0 000009  
167 000015 247 01 0 0 000016  
168 000016 254 00 0 0 000022  
169 000021 261 02 0 0 000001  
170 000022 247 02 0 0 000016  
171 000021 254 00 0 0 000015  
172 000022 051 14 0 0 000000  
173 000023 255 00 0 0 000000  
174  
175  
176 000024 051 03 0 0 000025  
177 000025 403 03 0 0 000004  
178 000026 260 02 0 0 000023  
179 000027 474 05 0 0 000009  
180 000093 051 04 0 0 000014  
181 000031 306 01 0 0 000040  
182 000032 254 00 0 0 000030  
183 000033 302 01 0 0 000033  
184 000034 306 01 0 0 000129  
185 000035 254 00 0 0 000129  
186 000036 302 01 0 0 000015  
187 000037 254 03 0 0 000051  
188 000040 251 04 0 0 000001  
189 000041 254 00 0 0 000014  
190 000042 051 04 0 0 000014  
191 000043 302 01 0 0 000015  
192 000044 306 01 0 0 000015  
193 000045 254 00 0 0 000105  
194 000046 306 01 0 0 000059  
195 000047 254 00 0 0 000105  
196 000051 301 01 0 0 000111  
197 000051 303 01 0 0 000132  
198 000052 254 00 0 0 000061  
199 000053 301 03 0 0 000006  
200 000054 254 00 0 0 000142  
201 000055 371 01 0 0 000048  
202 000056 136 01 0 0 000002  
203 000057 242 05 0 0 000000  
204 000080 344 03 0 0 000042  
205 000081 251 11 0 0 000000  
206 000061 051 03 0 0 000241  
207 000062 251 03 0 0 000014  
208 000063 254 00 0 0 000014  
209 000064  
210 000065  
211  
212  

SECTION C. INPUT THE FIRST WORD IN A LEARN SYSTEM COMMAND.

ENTER COMMAND *J

SETB C,D

MOVE B,POINDEX 6,0

SET E,

INPUT CHARACTER

CAIN A,40

IS CHARACTER A SPACE?

JRST -,2

YES, GO NEXT CHARACTER

CAIE A,33

USER TYPE "ESC"

JRST CMD2

YES, GO INTO CALC MODE (SECTION D)

CAIE A,15

USER TYPE "CARRIAGE RETURN"

JRST CMD2

NO, GO TO INPUT COMMAND.

TCALL 4,1

INPUT LINE FEED

JRST CMD

TDCALL 4,1

INPUT NEXT CHARACTER IN COMMAND STRING

JRST CMDIN

IF END OF WORD, GO TO CMDIN (SECTION D)

CAIE A,10

IF END OF Word, GO TO CMDIN (SECTION D)

CMDIN C,101

IF CHARACTER IS NOT ALPHABETIC, GO TYPE ERROR MESSAGE

JRST CMDERR

HAVE MORE THAN 8 CHARACTERS BEEN TYPED?

CAIE C,6

YES, IGNORE THIS CHARACTER

ADDI A,40

CONVERT CHARACTER TO SIXBIT CODE

IDP A,B

PLACE CHARACTER IN AC-D

LSH E,6

SHIFT CHARACTER DELETION MASK ONE CHARACTER RIGHT

ADJA C,CMD1

INCREMENT CHARACTER COUNT AND GO TO CMD1

CMDERR TDCALL 11,0

TDCALL 3,ASC12 /

ILLEGAL COMMAND, TRY AGAIN.

JRST CMD

SECTION D. INTERPRET FIRST WORD IN A LEARN SYSTEM COMMAND.
NO ERRORS DETECTED

PROGRAM BREAK IS 000386

3K CORE USED
TITLE PERL

COMMENT %

WRITTEN BY GARRET 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 I), LOCATED IN SECTION D

FILES - FROM LEDIT,MAC (SECTION I), LOCATED IN SECTION A

LST - FROM LEDIT,MAC (SECTION I), LOCATED IN SECTION C

RENAME - FROM LEDIT,MAC (SECTION I), LOCATED IN SECTION B

EXTERNAL SUBROUTINES CALLED BY PERL.MAC

IIT - INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC=8, LOCATED IN AUX2,MAC (SECTION M)

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)

IODB - 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)
SECTION A. EXECUTE A "FILES" COMMAND. THE FILES COMMAND

CAUSES A LIST OF LEARN FILES TO BE PRINTED ON THE USER'S

TERMINAL.
CALL A, 60

CALL A, 71

JST NEWNM, X

NEWMN1

SETB C,E

MOVE B, [POINT 6, E]

JST *2

INPT21

INPUT NEXT CHARACTER IN NEW FILENAME

INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN2

CURRENT CHARACTER AN "m"?

JST INPB

YES, GO TO INPB

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST INPB

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE (SECTION D)

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

CURRENT CHARACTER AN "m"?

JST INPB

YES, GO TO INPB

TTCALL 4,A

INPUT NEXT CHARACTER IN COMMAND STRING

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE (SECTION D)

INPT11

TTCALL 4,A

INPUT NEXT CHARACTER IN OLD FILENAME

CAIE A, 15

CAIE A, 49

JST NEWNM

YES, GO TO NEWNM

CALL A, 71

YES, GO INPUT NEXT CHARACTER

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

INPT11

TTCALL 4,A

INPUT NEXT CHARACTER IN OLD FILENAME

CAIE A, 15

CAIE A, 49

JST NEWNM

YES, GO TO NEWNM

CALL A, 71

YES, GO INPUT NEXT CHARACTER

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE

AOJA C, INPT2

INCREMENT CHARACTER COUNT AND GO TO INPT2

NAMIN1

TTCALL 4,A

INPUT NEXT CHARACTER

CALL A, 49

YES, GO INPUT NEXT CHARACTER

JST *2

YES, GO TO INPB

JST NMERR

YES, GO TYPE ERROR MESSAGE
SECTION C. EXECUTE THE LIST COMMAND. THE LIST COMMAND PREPARES A LISTING OF AN UNPROCESSED LESSON FILE AND PLACES IT IN PRINT.
<table>
<thead>
<tr>
<th>PERL</th>
<th>MACRO</th>
<th>47(204)-9 08127 2-AUG-74 PAGE 1-11</th>
</tr>
</thead>
</table>

572 083624
573 083625
574 083626
575 083627
576 083630
577 083631
578 083632
579 083633
580
581 083634
582 083635
583 083636
584 083637
585 083638
586 083639
587 083640
588 083641
589 083642
590 083643
591 083644
592 083645
593 083646
594 083647
595 083648
596 083649
597 083650
598 083651
599 083652
600 083653
601 083654
602 083655
603 083656
604
605
606 083657
607
608 083658
609 083659
610 083660
611 083661
612 083662
613 083663
614 083664
615 083665
616 083666
617 083667
618 083668

**JSP Q.OUTCHR**
MOVE A,F
JSP Q.IOGU
OUTPUT DAY

**JSP Q.IOGD**
MOVE A,G
JSP Q.IOOD
OUTPUT YEAR

**ASCIE /**
TIME /

**KSTIME A,**
IDIV A, *068000*
IDIV A, *06A*
MOVE F,B
JSP Q.IOGD
OUTPUT HOUR

**JSP Q.IOOD**
MOVE A,F
JSP Q.IOOD
OUTPUT MINUTE

**PPNI /**

**HLRE A,PPN**
JSP Q.IOOD
OUTPUT PROJECT NO. OF USER

**JSP Q.IOOD**
MOVE A, "I"
JSP Q.OUTCHR
OUTPUT PROGRAMMER NO. OF USER

**ASCIE /**

**MOVEI I:1**
SET PAGE COUNT TO 1

**MOVEI H:15**
SET LINE COUNT TO 13

**MOVEI M:15**
SET PAGE COUNT TO 1

**MOVEI K:A**
SAVE CHARACTER IN AC-K

**JSP Q.INCHR**
INPUT CHARACTER FROM UNPROCESSED LESSON

**AQS G**
INCREMENT CHARACTER COUNT

**CIN A,0**
END OF LESSON

**JST EOL**
YES, GO TO EOL

**CAIN A,36**
BEGINNING OF FRAME?

**AQJA F,FRAME**
YES, INCREMENT FRAME COUNT AND GO TO FRAME
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>ONAM</th>
<th>95#</th>
<th>113</th>
<th>249</th>
<th>251</th>
<th>252</th>
<th>254</th>
<th>256</th>
<th>258</th>
<th>265</th>
<th>275</th>
<th>529</th>
<th>531</th>
<th>532</th>
<th>533</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPN</td>
<td>152#</td>
<td>162</td>
<td>164</td>
<td>206</td>
<td>220</td>
<td>222</td>
<td>224</td>
<td>226</td>
<td>228</td>
<td>230</td>
<td>232</td>
<td>234</td>
<td>236</td>
<td>238</td>
</tr>
<tr>
<td>QUTCN</td>
<td>96#</td>
<td>238</td>
<td>240</td>
<td>242</td>
<td>244</td>
<td>246</td>
<td>248</td>
<td>250</td>
<td>252</td>
<td>254</td>
<td>256</td>
<td>258</td>
<td>260</td>
<td>262</td>
</tr>
<tr>
<td>OUTCN</td>
<td>96#</td>
<td>123</td>
<td>125</td>
<td>127</td>
<td>129</td>
<td>131</td>
<td>133</td>
<td>135</td>
<td>137</td>
<td>139</td>
<td>141</td>
<td>143</td>
<td>145</td>
<td>147</td>
</tr>
<tr>
<td>PAGE</td>
<td>620</td>
<td>640</td>
<td>642</td>
<td>644</td>
<td>646</td>
<td>648</td>
<td>650</td>
<td>652</td>
<td>654</td>
<td>656</td>
<td>658</td>
<td>660</td>
<td>662</td>
<td>664</td>
</tr>
<tr>
<td>PCL</td>
<td>96#</td>
<td>121</td>
<td>233</td>
<td>237</td>
<td>245</td>
<td>247</td>
<td>249</td>
<td>251</td>
<td>253</td>
<td>255</td>
<td>257</td>
<td>259</td>
<td>261</td>
<td>263</td>
</tr>
<tr>
<td>POINT</td>
<td>12#</td>
<td>218</td>
<td>220</td>
<td>222</td>
<td>224</td>
<td>226</td>
<td>228</td>
<td>230</td>
<td>232</td>
<td>234</td>
<td>236</td>
<td>238</td>
<td>240</td>
<td>242</td>
</tr>
<tr>
<td>PRT2</td>
<td>15#</td>
<td>257</td>
<td>267</td>
<td>277</td>
<td>287</td>
<td>297</td>
<td>307</td>
<td>317</td>
<td>327</td>
<td>337</td>
<td>347</td>
<td>357</td>
<td>367</td>
<td>377</td>
</tr>
<tr>
<td>PPN</td>
<td>96#</td>
<td>170</td>
<td>597</td>
<td>601</td>
<td>603</td>
<td>605</td>
<td>607</td>
<td>609</td>
<td>611</td>
<td>613</td>
<td>615</td>
<td>617</td>
<td>619</td>
<td>621</td>
</tr>
<tr>
<td>PRS</td>
<td>399</td>
<td>406#</td>
<td>413#</td>
<td>420#</td>
<td>427#</td>
<td>434#</td>
<td>441#</td>
<td>448#</td>
<td>455#</td>
<td>462#</td>
<td>469#</td>
<td>476#</td>
<td>483#</td>
<td>490#</td>
</tr>
<tr>
<td>PRT3</td>
<td>818</td>
<td>829#</td>
<td>836#</td>
<td>843#</td>
<td>850#</td>
<td>857#</td>
<td>864#</td>
<td>871#</td>
<td>878#</td>
<td>885#</td>
<td>892#</td>
<td>899#</td>
<td>906#</td>
<td>913#</td>
</tr>
<tr>
<td>Q</td>
<td>113#</td>
<td>121</td>
<td>233</td>
<td>245</td>
<td>257</td>
<td>269</td>
<td>281</td>
<td>293</td>
<td>305</td>
<td>317</td>
<td>329</td>
<td>341</td>
<td>353</td>
<td>365</td>
</tr>
<tr>
<td>REC</td>
<td>397</td>
<td>414#</td>
<td>207#</td>
<td>220#</td>
<td>233#</td>
<td>246#</td>
<td>259#</td>
<td>272#</td>
<td>285#</td>
<td>298#</td>
<td>311#</td>
<td>324#</td>
<td>337#</td>
<td>350#</td>
</tr>
<tr>
<td>RECS</td>
<td>168</td>
<td>198</td>
<td>297#</td>
<td>320#</td>
<td>343#</td>
<td>366#</td>
<td>389#</td>
<td>412#</td>
<td>435#</td>
<td>458#</td>
<td>481#</td>
<td>504#</td>
<td>527#</td>
<td>550#</td>
</tr>
<tr>
<td>RENAH</td>
<td>93</td>
<td>310#</td>
<td>333#</td>
<td>356#</td>
<td>379#</td>
<td>402#</td>
<td>425#</td>
<td>448#</td>
<td>471#</td>
<td>494#</td>
<td>517#</td>
<td>540#</td>
<td>563#</td>
<td>586#</td>
</tr>
<tr>
<td>RSLCTN</td>
<td>814</td>
<td>899#</td>
<td>984#</td>
<td>1069</td>
<td>1154</td>
<td>1239</td>
<td>1324</td>
<td>1409</td>
<td>1494</td>
<td>1579</td>
<td>1664</td>
<td>1749</td>
<td>1834</td>
<td>1919</td>
</tr>
<tr>
<td>SLASH</td>
<td>156#</td>
<td>288</td>
<td>320#</td>
<td>352#</td>
<td>384#</td>
<td>416#</td>
<td>448#</td>
<td>480#</td>
<td>512#</td>
<td>544#</td>
<td>576#</td>
<td>608#</td>
<td>640#</td>
<td>672#</td>
</tr>
<tr>
<td>STORE</td>
<td>96#</td>
<td>466</td>
<td>508</td>
<td>550</td>
<td>592</td>
<td>634</td>
<td>676</td>
<td>718</td>
<td>760</td>
<td>802</td>
<td>844</td>
<td>886</td>
<td>928</td>
<td>970</td>
</tr>
<tr>
<td>SWIN</td>
<td>357</td>
<td>367</td>
<td>377#</td>
<td>387#</td>
<td>397#</td>
<td>407#</td>
<td>417#</td>
<td>427#</td>
<td>437#</td>
<td>447#</td>
<td>457#</td>
<td>467#</td>
<td>477#</td>
<td>487#</td>
</tr>
<tr>
<td>SWIN2</td>
<td>374#</td>
<td>384#</td>
<td>394#</td>
<td>404#</td>
<td>414#</td>
<td>424#</td>
<td>434#</td>
<td>444#</td>
<td>454#</td>
<td>464#</td>
<td>474#</td>
<td>484#</td>
<td>494#</td>
<td>504#</td>
</tr>
<tr>
<td>TRANSF</td>
<td>824</td>
<td>828</td>
<td>832</td>
<td>836#</td>
<td>840#</td>
<td>844#</td>
<td>848#</td>
<td>852#</td>
<td>856#</td>
<td>860#</td>
<td>864#</td>
<td>868#</td>
<td>872#</td>
<td>876#</td>
</tr>
<tr>
<td>UNPFIL</td>
<td>816</td>
<td>829#</td>
<td>833#</td>
<td>837#</td>
<td>841#</td>
<td>845#</td>
<td>849#</td>
<td>853#</td>
<td>857#</td>
<td>861#</td>
<td>865#</td>
<td>869#</td>
<td>873#</td>
<td>877#</td>
</tr>
<tr>
<td>UNPS</td>
<td>181</td>
<td>187#</td>
<td>299#</td>
<td>303#</td>
<td>307#</td>
<td>311#</td>
<td>315#</td>
<td>319#</td>
<td>323#</td>
<td>327#</td>
<td>331#</td>
<td>335#</td>
<td>339#</td>
<td>343#</td>
</tr>
<tr>
<td>WROXT</td>
<td>240</td>
<td>299#</td>
<td>343#</td>
<td>347#</td>
<td>351#</td>
<td>355#</td>
<td>359#</td>
<td>363#</td>
<td>367#</td>
<td>371#</td>
<td>375#</td>
<td>379#</td>
<td>383#</td>
<td>387#</td>
</tr>
</tbody>
</table>
TITLE LPRINT

COMMENT X

PREPARED BY GARRET A. VANDER LUGT  COMPLETED 31-JUL-73

PART OF LEDIT.SHR VERSION 3

SPECIAL NOTE: THIS SUBROUTINE IS A MODIFIED VERSION OF A SUB-
ROUTINE WHICH WAS WRITTEN BY NORMAN O. 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 (#OPBLK) - ARGUMENTS FOR OPEN UFO (3 WORDS)

LINK+5 (#FILNAM) - FILENAME OF PARAMETER FILE

LINK+11 (#OLIST) - OUTPUT COMMAND LIST (2 WORDS)

LINK+13 (#OLSPEC) - EXTENDED LOOKUP ARGUMENTS (14 WORDS)

ONAM = 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(264)= # 00129 2-AUG-74 PAGE 1-1
LPRINT MAC 6-JUN-74

; 54 000067  WISEG
; 56 000068  ENTRY LPRINT
; 57 000069  EXTERN LINK,ONAM,SPACE,STORE
; 59 00006A  THSD=LINK
; 62 00006C  QUDIR=LINK+1
; 63 00006D  GBNLX=LINK+2
; 65 00006F  FMD=LINK+3
; 65 000070  OLISP=LINK+4
; 66 000071  PARMA=SPACE
; 67 000072  A=1
; 69 000073  B=2
; 70 000074  C=3
; 71 000075  D=4
; 73 000076  E=5
; 74 000077  F=6
; 75 000078  G=7
; 76 000079  H=8
; 77 00007A  ?=9

SECTION A: SET UP QUEUING PARAMETERS IN CORE:

16 00007B  OPN1  00007C  OPN2  00007D  OPN3  00007E  OPN4
2 00007F  SIXBIT /DSK/

000080  MOVE A, C34,113
LPRINT GETTAB A, GET MONITOR VERSION NO.

000084  HALT
000085  HRRES A
000086  CAGE A, EOF 0 0 0 0
000087  HALT
000088  INQ, HALT
000089  OPEN 0, OPN
00008A  HALT
00008B  MOVEI A, 40
00008C  SETEM PARMA, 1=A
00008D  SORU A=?
00008E  MOVEI A, 16
00008F  MOVEI A, GLSPEC
000090  MOVEI A, HLSPEC+1 MOVE 14 TO HLSPEC FOR EXTENDED LOOKUP
000091  SETEM HLSPEC+1 MOVE IT TO LOOKUP EFFECTIVE ADDRESS
000092  MOVEI A, PARMA+3 MOVE IT TO PARAMETER LIST ALSO
000093  MOVEI A, LIST
000094  MOVEI A, HLSPEC+3 MOVE IT TO PARAMETER LIST
000095  MOVEI A, HLSPEC+3
000096  MOVEI A, HLSPEC+3
000097  MOVEI A, HLSPEC+3
000098  MOVEI A, HLSPEC+3
000099  MOVEI A, HLSPEC+3
00009A  MOVEI A, HLSPEC+3
00009B  MOVEI A, HLSPEC+3
00009C  MOVEI A, HLSPEC+3
00009D  MOVEI A, HLSPEC+3
00009E  MOVEI A, HLSPEC+3
00009F  MOVEI A, HLSPEC+3
0000A0  MOVEI A, HLSPEC+3
0000A1  MOVEI A, HLSPEC+3
0000A2  MOVEI A, HLSPEC+3
0000A3  MOVEI A, HLSPEC+3
0000A4  MOVEI A, HLSPEC+3
0000A5  MOVEI A, HLSPEC+3
0000A6  MOVEI A, HLSPEC+3
0000A7  MOVEI A, HLSPEC+3
0000A8  MOVEI A, HLSPEC+3
0000A9  MOVEI A, HLSPEC+3
0000AA  MOVEI A, HLSPEC+3
0000AB  MOVEI A, HLSPEC+3
0000AC  MOVEI A, HLSPEC+3
0000AD  MOVEI A, HLSPEC+3
0000AE  MOVEI A, HLSPEC+3
0000AF  MOVEI A, HLSPEC+3
0000B0  MOVEI A, HLSPEC+3
0000B1  MOVEI A, HLSPEC+3
0000B2  MOVEI A, HLSPEC+3
0000B3  MOVEI A, HLSPEC+3
0000B4  MOVEI A, HLSPEC+3
0000B5  MOVEI A, HLSPEC+3
0000B6  MOVEI A, HLSPEC+3
0000B7  MOVEI A, HLSPEC+3
0000B8  MOVEI A, HLSPEC+3
0000B9  MOVEI A, HLSPEC+3
0000BA  MOVEI A, HLSPEC+3
0000BB  MOVEI A, HLSPEC+3
0000BC  MOVEI A, HLSPEC+3
0000BD  MOVEI A, HLSPEC+3
0000BE  MOVEI A, HLSPEC+3
0000BF  MOVEI A, HLSPEC+3
0000C0  MOVEI A, HLSPEC+3
0000C1  MOVEI A, HLSPEC+3
0000C2  MOVEI A, HLSPEC+3
0000C3  MOVEI A, HLSPEC+3
0000C4  MOVEI A, HLSPEC+3
0000C5  MOVEI A, HLSPEC+3
0000C6  MOVEI A, HLSPEC+3
0000C7  MOVEI A, HLSPEC+3
0000C8  MOVEI A, HLSPEC+3
0000C9  MOVEI A, HLSPEC+3
0000CA  MOVEI A, HLSPEC+3
0000CB  MOVEI A, HLSPEC+3
0000CC  MOVEI A, HLSPEC+3
0000CD  MOVEI A, HLSPEC+3
0000CE  MOVEI A, HLSPEC+3
0000CF  MOVEI A, HLSPEC+3
0000D0  MOVEI A, HLSPEC+3
0000D1  MOVEI A, HLSPEC+3
0000D2  MOVEI A, HLSPEC+3
0000D3  MOVEI A, HLSPEC+3
0000D4  MOVEI A, HLSPEC+3
0000D5  MOVEI A, HLSPEC+3
0000D6  MOVEI A, HLSPEC+3
0000D7  MOVEI A, HLSPEC+3
0000D8  MOVEI A, HLSPEC+3
0000D9  MOVEI A, HLSPEC+3
0000DA  MOVEI A, HLSPEC+3
0000DB  MOVEI A, HLSPEC+3
0000DC  MOVEI A, HLSPEC+3
0000DD  MOVEI A, HLSPEC+3
0000DE  MOVEI A, HLSPEC+3
0000DF  MOVEI A, HLSPEC+3
0000E0  MOVEI A, HLSPEC+3
0000E1  MOVEI A, HLSPEC+3
0000E2  MOVEI A, HLSPEC+3
0000E3  MOVEI A, HLSPEC+3
0000E4  MOVEI A, HLSPEC+3
0000E5  MOVEI A, HLSPEC+3
0000E6  MOVEI A, HLSPEC+3
0000E7  MOVEI A, HLSPEC+3
0000E8  MOVEI A, HLSPEC+3
0000E9  MOVEI A, HLSPEC+3
0000EA  MOVEI A, HLSPEC+3
0000EB  MOVEI A, HLSPEC+3
0000EC  MOVEI A, HLSPEC+3
0000ED  MOVEI A, HLSPEC+3
0000EE  MOVEI A, HLSPEC+3
0000EF  MOVEI A, HLSPEC+3
0000F0  MOVEI A, HLSPEC+3
0000F1  MOVEI A, HLSPEC+3
0000F2  MOVEI A, HLSPEC+3
0000F3  MOVEI A, HLSPEC+3
0000F4  MOVEI A, HLSPEC+3
0000F5  MOVEI A, HLSPEC+3
0000F6  MOVEI A, HLSPEC+3
0000F7  MOVEI A, HLSPEC+3
0000F8  MOVEI A, HLSPEC+3
0000F9  MOVEI A, HLSPEC+3
0000FA  MOVEI A, HLSPEC+3
0000FB  MOVEI A, HLSPEC+3
0000FC  MOVEI A, HLSPEC+3
0000FD  MOVEI A, HLSPEC+3
0000FE  MOVEI A, HLSPEC+3
0000FF  MOVEI A, HLSPEC+3
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>002114</td>
<td>JOB B, SET USER'S JOB NO.</td>
</tr>
<tr>
<td>161</td>
<td>002115</td>
<td>MOVE B, THISJOB, STORE IT IN THISJOB</td>
</tr>
<tr>
<td>162</td>
<td>002116</td>
<td>LRI A, (B)</td>
</tr>
<tr>
<td>163</td>
<td>002117</td>
<td>LRI A, 33</td>
</tr>
<tr>
<td>164</td>
<td>002118</td>
<td>GETTAB A,</td>
</tr>
<tr>
<td>165</td>
<td>002119</td>
<td>SETA A</td>
</tr>
<tr>
<td>166</td>
<td>002120</td>
<td>MOV A, PARAM+15</td>
</tr>
<tr>
<td>167</td>
<td>002121</td>
<td>LRI A, (B)</td>
</tr>
<tr>
<td>168</td>
<td>002122</td>
<td>LRI A, 31</td>
</tr>
<tr>
<td>169</td>
<td>002123</td>
<td>GETTAB A,</td>
</tr>
<tr>
<td>170</td>
<td>002124</td>
<td>SETA A</td>
</tr>
<tr>
<td>171</td>
<td>002125</td>
<td>MOV A, PARAM+16</td>
</tr>
<tr>
<td>172</td>
<td>002126</td>
<td>LRI A, (B)</td>
</tr>
<tr>
<td>173</td>
<td>002127</td>
<td>LRI A, 32</td>
</tr>
<tr>
<td>174</td>
<td>002128</td>
<td>GETTAB A,</td>
</tr>
<tr>
<td>175</td>
<td>002129</td>
<td>SETA A</td>
</tr>
<tr>
<td>176</td>
<td>002130</td>
<td>MOV A, PARAM+17</td>
</tr>
</tbody>
</table>

**SECTION B: WRITE PARAMETER FILE ON THE DISK.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>178</td>
<td>002131</td>
<td>MOVE A, [15, 16]</td>
</tr>
<tr>
<td>179</td>
<td>002132</td>
<td>GETTAB A,</td>
</tr>
<tr>
<td>180</td>
<td>002133</td>
<td>MOVSI A, GTMK</td>
</tr>
<tr>
<td>181</td>
<td>002134</td>
<td>MOV B, [14, 15]</td>
</tr>
<tr>
<td>182</td>
<td>002135</td>
<td>GETTAB B,</td>
</tr>
<tr>
<td>183</td>
<td>002136</td>
<td>MOV B, [13, 14]</td>
</tr>
<tr>
<td>184</td>
<td>002137</td>
<td>MOV C, [13, 14]</td>
</tr>
<tr>
<td>185</td>
<td>002138</td>
<td>MOV C, [13, 14]</td>
</tr>
<tr>
<td>186</td>
<td>002139</td>
<td>MOV D, [11, 12]</td>
</tr>
<tr>
<td>187</td>
<td>002140</td>
<td>GETTAB D,</td>
</tr>
<tr>
<td>188</td>
<td>002141</td>
<td>MOV B, QEDIR</td>
</tr>
<tr>
<td>189</td>
<td>002142</td>
<td>MOV C, OPNBLK</td>
</tr>
<tr>
<td>190</td>
<td>002143</td>
<td>SETM OPNBLK2</td>
</tr>
<tr>
<td>191</td>
<td>002144</td>
<td>MOV A, OPNBLK+1</td>
</tr>
<tr>
<td>192</td>
<td>002145</td>
<td>OPN D, OPNBLK+2</td>
</tr>
<tr>
<td>193</td>
<td>002146</td>
<td>HALT</td>
</tr>
<tr>
<td>194</td>
<td>002147</td>
<td>MOVSI B, LP1</td>
</tr>
<tr>
<td>195</td>
<td>002148</td>
<td>WHERE B,</td>
</tr>
<tr>
<td>196</td>
<td>002149</td>
<td>SETB B</td>
</tr>
<tr>
<td>197</td>
<td>002150</td>
<td>MOV B, [11, 12]</td>
</tr>
<tr>
<td>198</td>
<td>002151</td>
<td>LRI B, LP1</td>
</tr>
<tr>
<td>199</td>
<td>002152</td>
<td>MOV B, PARAM+3</td>
</tr>
</tbody>
</table>

**IFIND SUITABLE NAME FOR PARAMETER FILE.**

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>002153</td>
<td>MOVE A, HOST</td>
</tr>
<tr>
<td>201</td>
<td>002154</td>
<td>IDIV A, 140</td>
</tr>
<tr>
<td>202</td>
<td>002155</td>
<td>MOV A, LP1</td>
</tr>
<tr>
<td>203</td>
<td>002156</td>
<td>MOVSI B, LP1</td>
</tr>
<tr>
<td>204</td>
<td>002157</td>
<td>WHERE A,</td>
</tr>
<tr>
<td>205</td>
<td>002158</td>
<td>MOVE D, A</td>
</tr>
<tr>
<td>206</td>
<td>002159</td>
<td>MOV A, D,</td>
</tr>
<tr>
<td>207</td>
<td>002160</td>
<td>MOVSI B,</td>
</tr>
<tr>
<td>208</td>
<td>002161</td>
<td>IDIV A, 140</td>
</tr>
<tr>
<td>209</td>
<td>002162</td>
<td>IDIV A, 140</td>
</tr>
<tr>
<td>210</td>
<td>002163</td>
<td>IDIV A, 140</td>
</tr>
<tr>
<td>211</td>
<td>002164</td>
<td>IDIV A, 140</td>
</tr>
<tr>
<td>212</td>
<td>002165</td>
<td>IDIV A, 140</td>
</tr>
</tbody>
</table>
LPRINT  MACRO 47(284)-9 08129  2-AUG-74 PAGE 1-4

213 000175  603 03 00 00 770000
214 000176  254 00 00 00 001721
215 000177  255 03 00 00 616545
216 000220  222 02 00 00 002205
217 000221  200 05 00 00 000001
218 000222  076 03 00 00 000202
219 000223  625 33 00 00 777777
220 000224  254 00 00 00 001661
221 000225  326 03 00 00 001661
222 000226  200 32 00 00 000025
223 000227  225 03 00 00 616545
224 000228  200 05 00 00 000201
225 000229  270 00 00 00 000008
226 000230  277 00 00 00 000002
227 000231  254 04 00 00 000000
228 000232  200 01 00 00 002254
229 000233  202 31 00 00 000011
230 000234  422 00 00 00 000012
231 000235  067 00 00 00 000011
232 000236  370 00 00 00 000000
233 000237  071 00 00 00 000000
234 000238  400 02 00 00 000000
235 000239  400 02 00 00 000000
236 000240  000000

SECTION C.  WAKE UP THE SPOOLING PROGRAM.

240 000241  352 02 00 00 000002
241 000242  514 03 00 00 000002
242 000243  541 03 00 00 000003
243 000244  047 03 00 00 000041
244 000245  254 00 00 17 000030
245 000246  352 03 00 00 000245
246 000247  254 00 00 00 000224
247 000248  200 03 00 00 000002
248 000249  047 03 00 00 000073
249 000250  254 00 00 17 000000
250 000251  254 00 00 00 000224
251 000252  000000
252 000253  000000

NOSYM

END

NO ERRORS DETECTED

PROGRAM BREAK IS 000236

3K CORE USED
WRITED MACRO 47(204)-7 02122 2-AUG-74 PAGE 1

1 TITLE WRITED
2 COMMENT %
3 WRITTEN BY GARRET A. VANDER LUST COMPLETED 30-MAY-73
4 PART OF LEEDIT.SHR, VERSION 3
5 WRITED.MAC PERFORMS ALL OF THE LESSON EDITING FUNCTIONS FOR THE
6 LEARN SYSTEM.
7 CONTENTS OF WRITED.MAC
8 SECTION A. CONTAINS ENTRY POINTS TO WRITED.MAC
9 SECTION B. INTERPRETS 1ST CHARACTER IN AN EDIT COMMAND.
10 SECTION C. EXECUTES "X" COMMANDS.
11 SECTION D. EXECUTES "A" COMMANDS.
12 SECTION E. EXECUTES "T" COMMANDS.
13 SECTION F. EXECUTES "D" COMMANDS.
14 SECTION G. EXECUTES "I" COMMANDS.
15 SECTION H. EXECUTES "R" COMMANDS.
16 SECTION I. EXECUTES "K" COMMANDS.
17 SECTION J. EXECUTES "C" AND "N" COMMANDS.
19 SECTION L. SUBROUTINE TO INITIALIZE SPACE LIST.
20 SECTION M. SUBROUTINE TO FILL EDITING BUFFER.
21 SECTION N. SUBROUTINE TO EMPTY EDITING BUFFER.
22 SECTION O. SUBROUTINE TO RETURN TO BEGINNING OF LESSON.
23 SECTION P. SUBROUTINE TO FIND FRAME TO BE EDITED.

GENERAL DESCRIPTION OF LEARN LESSON EDITOR

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.

<table>
<thead>
<tr>
<th>BEFORE DELETION</th>
<th>AFTER DELETION</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK*01 1</td>
<td>LINK*01 1</td>
</tr>
<tr>
<td>LINK*11 2</td>
<td>LINK*11 3</td>
</tr>
<tr>
<td>LINK*21 3</td>
<td>LINK*21 3</td>
</tr>
<tr>
<td>LINK*31 -1</td>
<td>LINK*41 -1</td>
</tr>
</tbody>
</table>

NOTICE THAT NO CHARACTERS ARE ACTUALLY REMOVED FROM THE EDITING BUFFER.

THE EXAMPLE BELOW SHOWS THE CHANGE IN THE LINK LIST WHICH WOULD OCCUR IF A LINE WAS INSERTED IN FRONT OF LINE 2.

<table>
<thead>
<tr>
<th>BEFORE INSERTION</th>
<th>AFTER INSERTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK*01 1</td>
<td>LINK*01 5</td>
</tr>
<tr>
<td>LINK*11 2</td>
<td>LINK*11 2</td>
</tr>
<tr>
<td>LINK*21 3</td>
<td>LINK*21 3</td>
</tr>
<tr>
<td>LINK*31 4</td>
<td>LINK*31 4</td>
</tr>
<tr>
<td>LINK*41 -1</td>
<td>LINK*41 -1</td>
</tr>
</tbody>
</table>

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 64 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 WRTE,MAC (ALL IN SECTION A)

EDIT - FROM LEDIT,MAC (SECTION 1) IN RESPONSE TO USER'S EDIT COMMAND

EDIT2 - FROM LEDIT,MAC (SECTION 1) IN RESPONSE TO USER'S TEST COMMAND

WRTE - FROM LEDIT,MAC (SECTION 1) IN RESPONSE TO USER'S WRITE COMMAND

EXTERNAL SUBROUTINES CALLED BY WRTE,MAC

EOLIN - INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL THE END OF A LINE IS REACHED, LOCATED IN AUX2,MAC (SECTION A)

IIOD - INPUTS AN INTEGER FROM THE DISK AND PLACES IT IN AC-B, LOCATED IN AUX2,MAC (SECTION 1)

IIT - INPUTS AN INTEGER FROM THE TELETYPE AND PLACES IT IN AC-B, LOCATED IN AUX2,MAC (SECTION 1)

IIT2 - INPUTS AN INTEGER FROM THE TELETYPE AND PLACES IT IN AC-B, THE 1ST DIGIT IS ASSUMED TO BE IN AC-A, LOCATED IN AUX2,MAC (SECTION 1)

INCHR - INPUTS A CHARACTER FROM THE DISK AND PLACES IT IN AC-A, LOCATED IN AUX2,MAC (SECTION 1)

IOD - OUTPUTS AN INTEGER IN AC-A TO THE DISK, LOCATED IN AUX2,MAC (SECTION 1)

IOT - OUTPUTS AN INTEGER IN AC-A TO THE TELETYPE, LOCATED IN AUX2,MAC (SECTION 1)

OUTCHR - OUTPUTS A CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX2,MAC (SECTION 1)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN LDLOW,MAC)

ARGMN - ARGUMENTS FOR GETSEG UUD (6 WORDS)
BUFFER - LESSON EDITING BUFFER (1024 WORDS)
FRMINC - CONTAINS FRAME INCREMENT NO.
FRMNUN - CONTAIN NO. OF FRAME CURRENTLY IN THE EDITING BUFFER
FRMTYP - CONTAINS LETTER INDICATING TYPE OF FRAME IN THE EDITING BUFFER
FSTFRM - 1ST FRAME NO. IN EDIT COMMAND IS STORED HERE
FSTLIN - 1ST LINE NO. IN EDIT COMMAND IS STORED HERE
IBUF - BUFFER HEADER FOR DISK INPUT (3 WORDS)
INAM - LOOKUP ARGUMENTS FOR DISK INPUT (4 WORDS)
INBUF - DISK INPUT BUFFER NO. 1 (131 WORDS)
INBUF2 - DISK INPUT BUFFER NO. 2 (131 WORDS)
LINFUS - LINE CONTAINING A STRING TO BE REPLACED IS STORED HERE (16 WORDS)
LINK - LINK LIST (64 WORDS)
LSTFRM - 2ND FRAME NO. IN EDIT COMMAND IS STORED HERE
LSTLIN - 2ND LINE NO. IN EDIT COMMAND IS STORED HERE
NXTRFM - CONTAINS NO. OF FRAME FOLLOWING FRAME IN THE EDITING BUFFER
OBUF - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)
OLDSTR - STRING WHICH IS TO BE REPLACED IS STORED HERE (3 WORDS)
ONAM - ENTER_ARGUMENTS FOR DISK OUTPUT (4 WORDS)
OUTBUF2 - DISK OUTPUT BUFFER NO. 2 (131 WORDS)
OUTBUF - DISK OUTPUT BUFFER NO. 1 (131 WORDS)
PC1 - SUBROUTINES STORE RETURN ADDRESS HERE
PC2 - SUBROUTINES STORE RETURN ADDRESS HERE
PC3 - SUBROUTINES STORE RETURN ADDRESS HERE
PC4 - SUBROUTINES STORE RETURN ADDRESS HERE
POLST - PUSH DOWN LIST USED TO STORE ASCII DIGITS OF A

POLST - PUSH DOWN LIST USED TO STORE ASCII DIGITS OF A
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)

<table>
<thead>
<tr>
<th>BIT 14</th>
<th>COPY</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT 17</td>
<td>NO EMPTY</td>
</tr>
<tr>
<td>BIT 23</td>
<td>APPEND INSIDE OF FRAME</td>
</tr>
<tr>
<td>BIT 24</td>
<td>NEW FRAME NO.</td>
</tr>
<tr>
<td>BIT 25</td>
<td>OLD FRAME NO.</td>
</tr>
<tr>
<td>BIT 26</td>
<td>LAST FRAME</td>
</tr>
<tr>
<td>BIT 27</td>
<td>QUIT</td>
</tr>
<tr>
<td>BIT 29</td>
<td>OLD LESSON</td>
</tr>
<tr>
<td>BIT 34</td>
<td>TEST</td>
</tr>
<tr>
<td>BIT 39</td>
<td>PROCESS</td>
</tr>
</tbody>
</table>

DESCRIPTION OF FLAGS STORED IN AC=8

<table>
<thead>
<tr>
<th>%</th>
</tr>
</thead>
</table>

WISEG

ENTRY, HRT, EOT, EDT2

EXTERN BUFFER, NXTFRM, STORE, FRMNC, ARGNT, IOB, IBUF, OBUF

EXTERN FRMN, FRMTP, IOT, POLST, INCWR, OUTCWR, ECLIN

EXTERN PC1, ONAM, INAM, LINK, PC2, SPACE, PC3, PC4, TRMLK

EXTERN FRMFR, LSTFRM, FSTLIN, LSTLIN, OLDSTR, LINBUF

EXTERN IIF, IIT, IIO, INBUF, OBUF, INBUF, OUTBUF

<table>
<thead>
<tr>
<th>FLAGS80</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
</tr>
<tr>
<td>000001</td>
</tr>
<tr>
<td>000002</td>
</tr>
<tr>
<td>000003</td>
</tr>
<tr>
<td>000004</td>
</tr>
<tr>
<td>000005</td>
</tr>
<tr>
<td>000006</td>
</tr>
<tr>
<td>000007</td>
</tr>
<tr>
<td>000008</td>
</tr>
<tr>
<td>000009</td>
</tr>
<tr>
<td>000010</td>
</tr>
<tr>
<td>000011</td>
</tr>
<tr>
<td>000012</td>
</tr>
<tr>
<td>000013</td>
</tr>
</tbody>
</table>

PROT: 17788

POINT1: POINT 6, INAM
SECTION A.
INPUT LESSON NAME AND FRAME INCREMENT NUMBER IF ANY.
THE LOW SEGMENT IS EXPANDED TO 2K. BUFFERS ARE ADDED TO THE DISK.
BUFFER RINGS, FRMNUN AND NXNUM ARE SET TO ZERO, AND THE LAST
FRAME FLAG IS CLEARED.

000011' 202 17 0 0 00000000
EDT2: MOVEM 0,PC1
;SAVE RETURN ADDRESS

TRO FLAGS,100
;SET OLD LESSON FLAG

MOVEI 8,12

MOVEI 8,FRMNING
;INITIALIZE FRAME INCREMENT TO 10

XRST

WRTI TRO FLAGS,100
;CLEAR OLD LESSON FLAG AND SKIP NEXT LINE

EDTI TRO FLAGS,100
;SET OLD LESSON FLAG

Move 0,PC1
;SAVE RETURN ADDRESS

MOVEI 8,12

MOVEI 8,FRMNING
;INITIALIZE FRAME INCREMENT TO 10

XRST

TTCall 4,A
;INPUT NEXT CHARACTER IN COMMAND STRING

CAlN A,40
;IS CHARACTER A SPACE?

JRST +2
;YES, GO INPUT NEXT CHARACTER

JRST A,10
;NO

JRST A,40
;IS CHARACTER A SPACE?

JRST A,10
;NO

JRST A,15
;END OF COMMAND STRING, GO TO NONAME

JRST A,60
;END OF COMMAND STRING, GO TO NONAME

JRST A,71
;END OF COMMAND STRING, GO TO NONAME

JRST A,15
;END OF COMMAND STRING, GO TO NONAME

TTCall 11,0
;RETURN TO LEDIT SUBPROGRAM

LESSON NAMES MAY NOT BEGIN WITH A NUMERAL

RETURN TO LEDIT SUBPROGRAM
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>319</td>
<td>002053</td>
<td>IDPS A,B</td>
</tr>
<tr>
<td>320</td>
<td>002054</td>
<td>IDLE</td>
</tr>
<tr>
<td>321</td>
<td>002055</td>
<td>IDPA</td>
</tr>
<tr>
<td>322</td>
<td>002056</td>
<td>IDPC</td>
</tr>
<tr>
<td>323</td>
<td>002057</td>
<td>IDPA</td>
</tr>
<tr>
<td>324</td>
<td>002058</td>
<td>IDPA</td>
</tr>
<tr>
<td>325</td>
<td>002059</td>
<td>IDPA</td>
</tr>
<tr>
<td>326</td>
<td>002060</td>
<td>IDPA</td>
</tr>
<tr>
<td>327</td>
<td>002061</td>
<td>IDPA</td>
</tr>
<tr>
<td>328</td>
<td>002062</td>
<td>IDPA</td>
</tr>
<tr>
<td>329</td>
<td>002063</td>
<td>IDPA</td>
</tr>
<tr>
<td>330</td>
<td>002064</td>
<td>IDPA</td>
</tr>
<tr>
<td>331</td>
<td>002065</td>
<td>IDPA</td>
</tr>
<tr>
<td>332</td>
<td>002066</td>
<td>IDPA</td>
</tr>
<tr>
<td>333</td>
<td>002067</td>
<td>IDPA</td>
</tr>
<tr>
<td>334</td>
<td>002068</td>
<td>IDPA</td>
</tr>
<tr>
<td>335</td>
<td>002069</td>
<td>IDPA</td>
</tr>
<tr>
<td>336</td>
<td>002070</td>
<td>IDPA</td>
</tr>
<tr>
<td>337</td>
<td>002071</td>
<td>IDPA</td>
</tr>
<tr>
<td>338</td>
<td>002072</td>
<td>IDPA</td>
</tr>
<tr>
<td>339</td>
<td>002073</td>
<td>IDPA</td>
</tr>
<tr>
<td>340</td>
<td>002074</td>
<td>IDPA</td>
</tr>
<tr>
<td>341</td>
<td>002075</td>
<td>IDPA</td>
</tr>
<tr>
<td>342</td>
<td>002076</td>
<td>IDPA</td>
</tr>
<tr>
<td>343</td>
<td>002077</td>
<td>IDPA</td>
</tr>
<tr>
<td>344</td>
<td>002078</td>
<td>IDPA</td>
</tr>
<tr>
<td>345</td>
<td>002079</td>
<td>IDPA</td>
</tr>
<tr>
<td>346</td>
<td>002080</td>
<td>IDPA</td>
</tr>
<tr>
<td>347</td>
<td>002081</td>
<td>IDPA</td>
</tr>
<tr>
<td>348</td>
<td>002082</td>
<td>IDPA</td>
</tr>
<tr>
<td>349</td>
<td>002083</td>
<td>IDPA</td>
</tr>
<tr>
<td>350</td>
<td>002084</td>
<td>IDPA</td>
</tr>
<tr>
<td>351</td>
<td>002085</td>
<td>IDPA</td>
</tr>
<tr>
<td>352</td>
<td>002086</td>
<td>IDPA</td>
</tr>
<tr>
<td>353</td>
<td>002087</td>
<td>IDPA</td>
</tr>
<tr>
<td>354</td>
<td>002088</td>
<td>IDPA</td>
</tr>
<tr>
<td>355</td>
<td>002089</td>
<td>IDPA</td>
</tr>
<tr>
<td>356</td>
<td>002090</td>
<td>IDPA</td>
</tr>
<tr>
<td>357</td>
<td>002091</td>
<td>IDPA</td>
</tr>
<tr>
<td>358</td>
<td>002092</td>
<td>IDPA</td>
</tr>
<tr>
<td>359</td>
<td>002093</td>
<td>IDPA</td>
</tr>
<tr>
<td>360</td>
<td>002094</td>
<td>IDPA</td>
</tr>
<tr>
<td>361</td>
<td>002095</td>
<td>IDPA</td>
</tr>
<tr>
<td>362</td>
<td>002096</td>
<td>IDPA</td>
</tr>
<tr>
<td>363</td>
<td>002097</td>
<td>IDPA</td>
</tr>
<tr>
<td>364</td>
<td>002098</td>
<td>IDPA</td>
</tr>
<tr>
<td>365</td>
<td>002099</td>
<td>IDPA</td>
</tr>
<tr>
<td>366</td>
<td>002100</td>
<td>IDPA</td>
</tr>
<tr>
<td>367</td>
<td>002101</td>
<td>IDPA</td>
</tr>
<tr>
<td>368</td>
<td>002102</td>
<td>IDPA</td>
</tr>
<tr>
<td>369</td>
<td>002103</td>
<td>IDPA</td>
</tr>
<tr>
<td>370</td>
<td>002104</td>
<td>IDPA</td>
</tr>
<tr>
<td>371</td>
<td>002105</td>
<td>IDPA</td>
</tr>
</tbody>
</table>
IAD1 INPUT AND OUTPUT BUFFERS TO DISK BUFFER RINGS
SETZM INBUF2
MOVE A, (XH & 251, INBUF2 + 1)
MOVE A, INBUF2 + 1
MOVE A, XH & 251, INBUF3 + 1
MOVE A, INBUF3 + 1
MOVE A, (XH & 251, OUTBUF2 + 1)
MOVE A, OUTBUF2 + 1
MOVE A, (XH & 251, OUTBUF1 + 1)
MOVE A, OUTBUF1 + 1

MOVS1 A, UNP 1
MOVM A, INAM + 1
SETZM INAM + 3
LOOKUP L INAM + 1
LOOKUP LESSON FILE
JST NTFN D IF LESSON NOT FOUND, GO TO NTFN D
JST FND IF LESSON FOUND, GO TO FND

NTFN D NUM FLAGS, 120 I IS OLD LESSON FLAG SET?
JST NTFN D YES, GO TYPE ERROR MESSAGE
JST NTER NO LESSON FILE
FN D NUM FLAGS, 120 I IS OLD LESSON FLAG SET?
JST OLDAM NO, GO TYPE ERROR MESSAGE

ENTER A, INAM
MOVM A, ONAM
MOVS1, A, UNP 1
MOVM A, ONAM + 1
MOVM A, PROT
MOVM A, ONAM + 2
MOVM A, ONAM + 3
SETM ONAM + 3
ENTER NEW LESSON FILE
JST NORDM ERROR RETURN, GO TYPE ERROR MESSAGE

TRF FLAGS, 1200 I CLEAR LAST FRAME FLAG
SETM FRNM 1 SET FRAM NO, TO ZERO
SETM NTFRM SET NEXT FRAME NO, TO ZERO
TRF FLAGS, 1 I CLEAR NO EMPTY FLAG
TRN FLAGS, 120 I IS THIS A NEW LESSON?
JST NEWLES YES, GO TO NEWLES

SECTION B. INPUT FIRST CHARACTER IN EDIT COMMAND STRING AND
BRANCH TO THE PROPER ROUTINE TO EXECUTE THE COMMAND.

GETCHI MOVE A, TMBLK MOVE ADDRESS OF TMOP, ARGUMENTS TO AC-A
TMOP, A, 1 ISKIP IF TTY OUTPUT BUFFER IS NOT EMPTY
GETPF C, I SLEEP A SEC BEFORE CHECKING BUFFER AGAIN
MOVE C, I 100 CHECK BUFFER AGAIN
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
<th>Memory Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>4) TN. P</td>
<td>000515</td>
<td>120 105 040 106 122</td>
</tr>
<tr>
<td>501</td>
<td>3) TN. P=Q</td>
<td>000516</td>
<td>101 115 105 123 040</td>
</tr>
<tr>
<td>502</td>
<td></td>
<td>000517</td>
<td>116 040 124 112 122</td>
</tr>
<tr>
<td>503</td>
<td></td>
<td>000520</td>
<td>125 040 115 019 012</td>
</tr>
<tr>
<td>504</td>
<td></td>
<td>000521</td>
<td>041 051 040 042 124</td>
</tr>
<tr>
<td>505</td>
<td></td>
<td>000522</td>
<td>116 040 124 011 011</td>
</tr>
<tr>
<td>506</td>
<td></td>
<td>000523</td>
<td>011 055 124 131 128</td>
</tr>
<tr>
<td>507</td>
<td></td>
<td>000524</td>
<td>105 040 114 111 116</td>
</tr>
<tr>
<td>508</td>
<td></td>
<td>000525</td>
<td>105 040 120 044 111</td>
</tr>
<tr>
<td>509</td>
<td></td>
<td>000526</td>
<td>116 040 106 122 101</td>
</tr>
<tr>
<td>510</td>
<td></td>
<td>000527</td>
<td>115 040 040 116 115</td>
</tr>
<tr>
<td>511</td>
<td></td>
<td>000530</td>
<td>012 065 051 040 040</td>
</tr>
<tr>
<td>512</td>
<td></td>
<td>000531</td>
<td>124 116 054 128 055</td>
</tr>
<tr>
<td>513</td>
<td></td>
<td>000532</td>
<td>121 011 011 011 055</td>
</tr>
<tr>
<td>514</td>
<td></td>
<td>000533</td>
<td>124 111 120 129 040</td>
</tr>
<tr>
<td>515</td>
<td></td>
<td>000534</td>
<td>114 111 116 105 123</td>
</tr>
<tr>
<td>516</td>
<td></td>
<td>000535</td>
<td>040 120 040 124 110</td>
</tr>
<tr>
<td>517</td>
<td></td>
<td>000536</td>
<td>122 125 040 121 040</td>
</tr>
<tr>
<td>518</td>
<td></td>
<td>000537</td>
<td>111 116 040 106 122</td>
</tr>
<tr>
<td>519</td>
<td></td>
<td>000540</td>
<td>101 115 105 040 116</td>
</tr>
<tr>
<td>520</td>
<td></td>
<td>000541</td>
<td>015 012 015 012 261</td>
</tr>
<tr>
<td>521</td>
<td></td>
<td>000542</td>
<td>051 040 040 133 116</td>
</tr>
<tr>
<td>522</td>
<td></td>
<td>000543</td>
<td>011 011 011 011 055</td>
</tr>
<tr>
<td>523</td>
<td></td>
<td>000544</td>
<td>103 110 101 116 107</td>
</tr>
<tr>
<td>524</td>
<td></td>
<td>000545</td>
<td>105 040 106 122 101</td>
</tr>
<tr>
<td>525</td>
<td></td>
<td>000546</td>
<td>115 105 040 111 114</td>
</tr>
<tr>
<td>526</td>
<td></td>
<td>000547</td>
<td>103 122 105 115 105</td>
</tr>
<tr>
<td>527</td>
<td></td>
<td>000550</td>
<td>116 124 040 124 117</td>
</tr>
<tr>
<td>528</td>
<td></td>
<td>000551</td>
<td>040 116 115 040 040</td>
</tr>
<tr>
<td>529</td>
<td></td>
<td>000552</td>
<td>012 061 091 040 040</td>
</tr>
<tr>
<td>530</td>
<td></td>
<td>000553</td>
<td>105 011 011 011 011</td>
</tr>
<tr>
<td>531</td>
<td></td>
<td>000554</td>
<td>055 105 130 111 124</td>
</tr>
<tr>
<td>532</td>
<td></td>
<td>000555</td>
<td>040 106 112 117 115</td>
</tr>
<tr>
<td>533</td>
<td></td>
<td>000556</td>
<td>040 124 112 105 120</td>
</tr>
<tr>
<td>534</td>
<td></td>
<td>000557</td>
<td>105 124 111 124 117</td>
</tr>
<tr>
<td>535</td>
<td></td>
<td>000558</td>
<td>122 015 012 062 051</td>
</tr>
<tr>
<td>536</td>
<td></td>
<td>000559</td>
<td>040 040 105 120 011</td>
</tr>
<tr>
<td>537</td>
<td></td>
<td>000560</td>
<td>011 011 011 055 055</td>
</tr>
<tr>
<td>538</td>
<td></td>
<td>000561</td>
<td>130 111 124 043 101</td>
</tr>
<tr>
<td>539</td>
<td></td>
<td>000562</td>
<td>116 124 040 122 122</td>
</tr>
<tr>
<td>540</td>
<td></td>
<td>000563</td>
<td>117 113 105 123 123</td>
</tr>
<tr>
<td>541</td>
<td></td>
<td>000566</td>
<td>040 114 105 123 123</td>
</tr>
<tr>
<td>542</td>
<td></td>
<td>000567</td>
<td>117 116 015 012 063</td>
</tr>
<tr>
<td>543</td>
<td></td>
<td>000570</td>
<td>051 040 040 105 74</td>
</tr>
<tr>
<td>544</td>
<td></td>
<td>000571</td>
<td>011 011 011 011 055</td>
</tr>
<tr>
<td>545</td>
<td></td>
<td>000572</td>
<td>105 122 111 124 054</td>
</tr>
<tr>
<td>546</td>
<td></td>
<td>000573</td>
<td>040 126 112 117 103</td>
</tr>
<tr>
<td>547</td>
<td></td>
<td>000574</td>
<td>105 123 123 054 040</td>
</tr>
<tr>
<td>548</td>
<td></td>
<td>000575</td>
<td>101 116 104 040 124</td>
</tr>
<tr>
<td>549</td>
<td></td>
<td>000576</td>
<td>105 123 124 040 114</td>
</tr>
<tr>
<td>550</td>
<td></td>
<td>000577</td>
<td>105 123 123 117 116</td>
</tr>
</tbody>
</table>
SECTION C. EXECUTE "X" COMMAND, STORE FRAME NUMBER INCREMENT IN FRMING.

00611 265 17 0 0 000655
JSP 0.117
INPUT INCREMENT NO.

00612 322 22 0 0 000624
JUMPE B,XERR
IF NO, IS ZERO, GO TYPE ERROR MESSAGE

00613 254 02 0 0 000615
JRST +2
GO CHECK 1ST CHARACTER AFTER NO.

00614 261 04 0 0 000601
TDCALL 4,A
IS CHARACTER A SPACE?

00615 336 01 0 0 000540
CAI H A,40
YES, GO INPUT NEXT CHARACTER

00616 254 02 0 0 000614
JRST -2
YES, GO INPUT NEXT CHARACTER

00617 332 01 0 0 000505
CAI H A,15
END OF COMMAND STRING?

00618 254 02 0 0 000524
JRST ILLCMD
NO, GO TYPE ERROR MESSAGE (SECTION B)

00619 202 02 0 0 000564
MOVEH B,FRING
STORE INCREMENT IN FRING

00620 231 04 0 0 000501
TDCALL 4,A
INPUT LINE FEED

00621 254 00 0 0 000512
JRST GETCMD
GO ACCEPT NEXT COMMAND (SECTION B)

00622 054 11 0 0 000200
XERR
TDCALL 11,0

SECTION D. EXECUTE "A" COMMAND. "A" COMMANDS ARE USED TO JADD LINES TO THE END OF A FRAME OR FRAMES TO THE END OF A LESSON.

00627 051 04 0 0 000081
APPND1
TDCALL 4,A
INPUT NEXT CHARACTER IN COMMAND STRING

00628 326 01 0 0 000348
CAI H A,43
IS CHARACTER A SPACE?

00629 254 00 0 0 000627
JRST -2
YES, GO INPUT NEXT CHARACTER

00630 365 01 0 0 000015
CAI H A,15
END OF COMMAND STRING?

00631 254 00 0 0 000666
JRST APPALL
YES, GO TO APPALL

00632 255 17 0 0 000000
JSP 0.11T2
INPUT FRAME NO.

00633 254 00 0 0 000637
JRST +2
GO CHECK 1ST CHARACTER AFTER NO.

00634 254 00 0 0 000631
TDCALL 4,A

00635 365 01 0 0 000240
CAI H A,48
IS CHARACTER A SPACE?

00636 254 02 0 0 000636
JRST -2
YES, GO INPUT NEXT CHARACTER

00637 362 01 0 0 000315
CAI H A,13
END OF COMMAND STRING?

00638 254 02 0 0 000642
JRST ILLCMD
NO, GO TYPE ERROR MESSAGE (SECTION B)

00639 202 02 0 0 000501
TDCALL 4,A
INPUT LINE FEED

00640 051 04 0 0 000081
MOVEH B,FSTFRM
STORE FRAME NO. IN FSTFRM

00641 254 17 0 0 000303
JSP G,FNDFFM
FIND FRAME TO WHICH LINES WILL BE APPENDED (SECTION P)

00642 660 00 0 0 010000
TRO FLAGS=10000
SET APPEND INSIDE OF FRAME FLAG
FIND BUFFER SLOT CONTAINING LAST LINE IN FRAME
SETZ G
MOVE E,1
SET LINE NO. TO 1
APPNL: TCALL 4,A
INPUT LINE FEED
SKIP FRNUM, IS THERE A FRAME IN THE EDITING BUFFER?
JSP Q,EMPTY YES, EMPTY EDITING BUFFER (SECTION N)
JRT ENDLES YES, GO TO ENDLES
SKIP NXTFRM, IS NEXT FRAME NO. BEEN READ?
JRT FNDEND NO, GO TO FNDEND
MOVE B,NXTFRM
MOVE B,FRNUM, MOVE NEXT FRAME NO. TO FRNUM
MOVE A,36
JSP Q,OUTCHR OUTPUT <36>
MOVE A,B
JSP Q,GO OUTPUT NO.
MOVE A,40
JSP Q,OUTCHR OUTPUT SPACE AFTER FRAME NO.
FDEND1: JSP Q,INCHR INPUT CHARACTER
CAIN A,4 ENDOF LESSON?
JRL ENDLES YES, GO TO ENDLES
CAIN A,36 BEGINNING OF FRAME?
JRT FNUM YES, GO TO FNUM
JSP Q,OUTCHR OUTPUT CHARACTER
FDEND JRT FNDEND

READ IN FRAME NO.
JSP Q,OUTCHR OUTPUT <36>
SET Z 0
JSP Q,INCHR INPUT CHARACTER
IIN1: JSP Q,OUTCHR OUTPUT CHARACTER
CAIN A,40 ENDOF FRAME NO.?
JRL NUMIN1 YES, GO STORE IT
SUB A,60
JRT IN1
IMUL B,12
ADD B,A
JRT IN1
NUMIN1: MOVE B,FRNUM ISTORE FRAME NO. IN FRNUM
JRT FNDEND

IDETERMINE NO. OF FRAME TO BE APPENDED MINUS INCREMENT
ENDLES IMOVE A,FRNUM
IDIV A,FRMNZ IDIVIDE NO. OF LAST FRAME BY FRAME INCREMENT
IMUL A,FRMNZ IMULTIPLY RESULT BY FRAME INCREMENT
MOVE A,FRNUM ISTORE NO. IN FRNUM
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>609</td>
<td>202724</td>
<td>NEWLEST TR2 FLAGS, 10000 CLEAR APPEND INSIDE OF FRAME FLAG</td>
</tr>
<tr>
<td>610</td>
<td>202725</td>
<td>APP21 TDCALL 3.CALF</td>
</tr>
<tr>
<td>611</td>
<td>03726</td>
<td>MOVE A, RMIN                                            INPUT FRAME INCREMENT IN AC-A</td>
</tr>
<tr>
<td>612</td>
<td>03727</td>
<td>ADDO A, FRNUM ADD TO GET NO. OF APPENDED FRAME</td>
</tr>
<tr>
<td>613</td>
<td>202730</td>
<td>IJF 0.0099 IS FRAME NO. GREATER THAN 999</td>
</tr>
<tr>
<td>614</td>
<td>03728</td>
<td>JUSTR FNE 7 YES, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>615</td>
<td>202731</td>
<td>APP3 TDCALL 3.CASC2 7 /7</td>
</tr>
<tr>
<td>616</td>
<td>03731</td>
<td>JSP F,1088 /TYPE FRAME NO.</td>
</tr>
<tr>
<td>617</td>
<td>03732</td>
<td>TDCALL 3.CASC2 /TYPE=7/7</td>
</tr>
<tr>
<td>618</td>
<td>03733</td>
<td>TDCALL 4,A INPUT FRAME TYPE</td>
</tr>
<tr>
<td>619</td>
<td>202735</td>
<td>CAIN A, 40 IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>620</td>
<td>03736</td>
<td>JUSTR 7 2 YES, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>621</td>
<td>03737</td>
<td>MOVE A, RMTYP STORE FRAME TYPE IN FRMTYP</td>
</tr>
<tr>
<td>622</td>
<td>202738</td>
<td>CAIE A, &quot;0&quot;</td>
</tr>
<tr>
<td>623</td>
<td>03739</td>
<td>CAIE A, &quot;0&quot;</td>
</tr>
<tr>
<td>624</td>
<td>03741</td>
<td>CAIN A, &quot;C&quot;</td>
</tr>
<tr>
<td>625</td>
<td>202743</td>
<td>JUSTR OK IF G OR C FRAME, GO TO OK</td>
</tr>
<tr>
<td>626</td>
<td>03744</td>
<td>CAIE A, &quot;B&quot;</td>
</tr>
<tr>
<td>627</td>
<td>03745</td>
<td>CAIN A, &quot;R&quot;</td>
</tr>
<tr>
<td>628</td>
<td>202746</td>
<td>JUSTR OK IF R OR B FRAME, GO TO OK</td>
</tr>
<tr>
<td>629</td>
<td>202747</td>
<td>CAIN A, 175</td>
</tr>
<tr>
<td>630</td>
<td>03748</td>
<td>JUSTR STOPA IF &quot;ESC&quot; GO TO STOPA</td>
</tr>
<tr>
<td>631</td>
<td>202749</td>
<td>TDCALL 11, 9</td>
</tr>
<tr>
<td>632</td>
<td>03750</td>
<td>TDCALL 3.CASC2</td>
</tr>
<tr>
<td>633</td>
<td>03751</td>
<td>FRAME TYPE MUST BE Q, G, R, or B.</td>
</tr>
<tr>
<td>634</td>
<td>03752</td>
<td></td>
</tr>
<tr>
<td>635</td>
<td>03753</td>
<td>OK1 TDCALL 4,A INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>636</td>
<td>03754</td>
<td>CAIN A, 40 IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>637</td>
<td>202751</td>
<td>JUSTR APPS</td>
</tr>
<tr>
<td>638</td>
<td>03752</td>
<td>TDCALL 11, 9</td>
</tr>
<tr>
<td>639</td>
<td>03753</td>
<td>TDCALL 3.CASC2 FRAME NUMBER MAY NOT EXCEED 999</td>
</tr>
<tr>
<td>640</td>
<td>03754</td>
<td></td>
</tr>
<tr>
<td>641</td>
<td>202757</td>
<td>JUSTR STOPA</td>
</tr>
<tr>
<td>642</td>
<td>03758</td>
<td>TDCALL 4,A INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>643</td>
<td>03759</td>
<td>CAIN A, 40 IS CHARACTER A SPACE?</td>
</tr>
<tr>
<td>644</td>
<td>03760</td>
<td>JUSTR 7 2 YES, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>645</td>
<td>03761</td>
<td>CAIE A, 15 END OF COMMAND STRING?</td>
</tr>
<tr>
<td>646</td>
<td>202762</td>
<td>JUSTR ERR GET, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>647</td>
<td>03763</td>
<td>TDCALL 4,A INPUT CHARACTER AT END OF LINE FEED</td>
</tr>
<tr>
<td>648</td>
<td>03764</td>
<td>JSP F, SETBUF INITIALIZE SPACE LIST (SECTION L)</td>
</tr>
<tr>
<td>649</td>
<td>03765</td>
<td>SET2 F,</td>
</tr>
<tr>
<td>650</td>
<td>202766</td>
<td>MOVE G, POINT2 REMOVE SLOT FROM SPACE LIST</td>
</tr>
<tr>
<td>651</td>
<td>03767</td>
<td>MOVE A, POINT2 PUT POINTER TO SLOT ZERO IN AC-G</td>
</tr>
<tr>
<td>652</td>
<td>03768</td>
<td>IDPS A, G PUT &lt;36&gt; IN EDITING BUFFER</td>
</tr>
<tr>
<td>653</td>
<td>03769</td>
<td>IPUT FRAME NO., IN SLOT ZERO OF EDITING BUFFER</td>
</tr>
<tr>
<td>654</td>
<td>03770</td>
<td>MOVE A, FRNUM GET FRAME NO. IN AC-A</td>
</tr>
<tr>
<td>655</td>
<td>03771</td>
<td></td>
</tr>
<tr>
<td>656</td>
<td>03772</td>
<td></td>
</tr>
<tr>
<td>657</td>
<td>03773</td>
<td></td>
</tr>
<tr>
<td>658</td>
<td>03774</td>
<td></td>
</tr>
<tr>
<td>659</td>
<td>03775</td>
<td></td>
</tr>
<tr>
<td>660</td>
<td>03776</td>
<td></td>
</tr>
<tr>
<td>661</td>
<td>03777</td>
<td></td>
</tr>
</tbody>
</table>
715 001061 326 01 00 000175  CAIN A.175  "IS CHARACTER AN "ESC"?"
716 001062 254 02 00 001057  JRST NXTCH  "YES, GO INPUT NEXT CHARACTER"
717 001063 156 01 00 000007  IDPB A.G  "NO, PUT CHARACTER IN BUFFER"
718 001064 305 01 00 0020612  CAIN A.12  "END OF LINE"
719 001065 344 05 00 001032  AOJA E.NXTLN  "YES, INCREMENT LINE NO. AND GO TO NXTLN"
720 001066 254 00 00 001067  JRST NXTCH  "NO, GO INPUT NEXT CHARACTER"
721 001067 211 22 00 000001  STOPF  "MOVNI B.1"
722 001068 202 32 00 0010514  MOVE M.BLINK (F)  "INPUT -1 IN LINK LIST TO INDICATE LAST LINE"
723 001069 862 00 00 010000  TRNE FLAGS.10000  "IS APPEND INSIDE OF FRAME FLAG SET?"
724 001070 221071 254 02 00 001067  JRST STOP2  "YES, GO TO STOP2"
725 001071 265 17 00 0022717  TDCALL 3.CRLF  "JSP Q.EMPTY  "EMPTY BUFFER (SECTION N)"
726 001072 265 17 00 0020725  JRST APP2  "YES, APPEND INSIDE OF FRAME FLAG SET?"
727 001073 265 17 00 0020601  STOPA1  "TL0 FLAGS.1  "SET NO EMPTY FLAG"
728 001074 265 17 00 010000  TDCALL 3.CRLF  "YES, SET LAST FRAME FLAG"
729 001075 265 17 00 0022735  JS0 Q.BEG  "GO TO BEGINNING OF LESSON (SECTION B)"
730 001076 251 03 00 000006  STOPA21  "TDCALL 3.CRLF  "JSP T.Getch  "GO ACCEPT NEXT COMMAND (SECTION B)"
731 001077 251 11 00 000000  BULLF  "TDCALL 11.0  "EDITING BUFFER FULL"
732 001078 251 11 00 000000  TDCALL 3.LASCIE  "WARNING! ONLY 64 LINES ARE ALLOWED IN A SINGLE FRAME"
733 001079 254 00 00 001036  JRST WNDON  "WARNING! COMMANDS CAUSE SELECTED FRAMES OR LINES TO BE TYPED ON THE USER'S TELETYPewriter."
<table>
<thead>
<tr>
<th>Line No.</th>
<th>Address</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1088</td>
<td>016020</td>
<td>AOS E</td>
</tr>
<tr>
<td>1087</td>
<td>016021</td>
<td>INCREMENT LINE NO.</td>
</tr>
<tr>
<td>1088</td>
<td>016022</td>
<td>AS 1,4, 2, FSTLIN</td>
</tr>
<tr>
<td>1088</td>
<td>016023</td>
<td>IS LINE TO BE INSERTED HERE?</td>
</tr>
<tr>
<td>1089</td>
<td>016024</td>
<td>JST INS</td>
</tr>
<tr>
<td>1089</td>
<td>016025</td>
<td>INO, GO TO INS</td>
</tr>
<tr>
<td>1089</td>
<td>016026</td>
<td>HCALL A, J</td>
</tr>
<tr>
<td>1090</td>
<td>016027</td>
<td>INSET NO. OF SLOTS LEFT IN SPACE LIST</td>
</tr>
<tr>
<td>1090</td>
<td>016028</td>
<td>JUME A,BFULL2</td>
</tr>
<tr>
<td>1090</td>
<td>016029</td>
<td>IF ALL SLOTS USED, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>1091</td>
<td>01602A</td>
<td>MOVE A, 40</td>
</tr>
<tr>
<td>1092</td>
<td>01602B</td>
<td>CAJE E, 12</td>
</tr>
<tr>
<td>1093</td>
<td>01602C</td>
<td>TYPE SPACE BEFORE LINE NO. LESS THAN TEN</td>
</tr>
<tr>
<td>1094</td>
<td>01602D</td>
<td>TDCALL 1, A</td>
</tr>
<tr>
<td>1095</td>
<td>01602E</td>
<td>MOVE A, E</td>
</tr>
<tr>
<td>1095</td>
<td>01602F</td>
<td>JSP Q, 10T</td>
</tr>
<tr>
<td>1096</td>
<td>016030</td>
<td>TDCALL 3, SPCAST</td>
</tr>
<tr>
<td>1097</td>
<td>016031</td>
<td>TDCALL 4, A</td>
</tr>
<tr>
<td>1098</td>
<td>016032</td>
<td>IINPUT 1ST CHARACTER IN LINE</td>
</tr>
<tr>
<td>1099</td>
<td>016033</td>
<td>CAJE A, 33</td>
</tr>
<tr>
<td>1100</td>
<td>016034</td>
<td>CTFL E, 12, 25, A</td>
</tr>
<tr>
<td>1101</td>
<td>016035</td>
<td>IS CHARACTER AN &quot;ESC&quot;?</td>
</tr>
<tr>
<td>1102</td>
<td>016036</td>
<td>JST INS</td>
</tr>
<tr>
<td>1102</td>
<td>016037</td>
<td>YES, GO TO INS</td>
</tr>
<tr>
<td>1103</td>
<td>016038</td>
<td>POP J, D</td>
</tr>
<tr>
<td>1103</td>
<td>016039</td>
<td>GET EMPTY SLOT FROM SPACE LIST</td>
</tr>
<tr>
<td>1104</td>
<td>01603A</td>
<td>MOVE D, LINK (G)</td>
</tr>
<tr>
<td>1104</td>
<td>01603B</td>
<td>MOVE SLOT TO LINK LIST</td>
</tr>
<tr>
<td>1104</td>
<td>01603C</td>
<td>MOVE C, D</td>
</tr>
<tr>
<td>1104</td>
<td>01603D</td>
<td>SAVE SLOT NO. IN AG-C</td>
</tr>
<tr>
<td>1105</td>
<td>01603E</td>
<td>MOVE C, D</td>
</tr>
<tr>
<td>1105</td>
<td>01603F</td>
<td>MOVE SLOT NO. TO AG-C</td>
</tr>
<tr>
<td>1106</td>
<td>016040</td>
<td>JSET UP BYTE POINTER TO PROPER SLOT</td>
</tr>
<tr>
<td>1107</td>
<td>016041</td>
<td>IMULC 1, 20</td>
</tr>
<tr>
<td>1108</td>
<td>016042</td>
<td>JEACH SLOT OCCUPIES 16 WORDS OF CORE</td>
</tr>
<tr>
<td>1109</td>
<td>016043</td>
<td>ADDIC C, BUFFER</td>
</tr>
<tr>
<td>1109</td>
<td>016044</td>
<td>ADDON BASE ADDRESS OF BUFFER</td>
</tr>
<tr>
<td>1110</td>
<td>016045</td>
<td>HLL C, POINT2</td>
</tr>
<tr>
<td>1111</td>
<td>016046</td>
<td>JST R, 4</td>
</tr>
<tr>
<td>1111</td>
<td>016047</td>
<td>LIST CHARACTER IS ALREADY IN</td>
</tr>
<tr>
<td>1112</td>
<td>016048</td>
<td>TTDCALL 4, A</td>
</tr>
<tr>
<td>1113</td>
<td>016049</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>1113</td>
<td>01604A</td>
<td>CAJE A, 33</td>
</tr>
<tr>
<td>1114</td>
<td>01604B</td>
<td>CTFL E, 12, 25, A</td>
</tr>
<tr>
<td>1114</td>
<td>01604C</td>
<td>IS CHARACTER AN &quot;ESC&quot;?</td>
</tr>
<tr>
<td>1115</td>
<td>01604D</td>
<td>JST INS</td>
</tr>
<tr>
<td>1115</td>
<td>01604E</td>
<td>YES, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>1116</td>
<td>01604F</td>
<td>IDPB A, C</td>
</tr>
<tr>
<td>1117</td>
<td>016050</td>
<td>INPUT CHARACTER IN EDITING BUFFER</td>
</tr>
<tr>
<td>1117</td>
<td>016051</td>
<td>CAJE A, 12</td>
</tr>
<tr>
<td>1117</td>
<td>016052</td>
<td>INEND OF LINE</td>
</tr>
<tr>
<td>1118</td>
<td>016053</td>
<td>ADEJ E, INSB</td>
</tr>
<tr>
<td>1118</td>
<td>016054</td>
<td>YES, INCREMENT LINE NO. AND GO TO INSB</td>
</tr>
<tr>
<td>1119</td>
<td>016055</td>
<td>JST INS</td>
</tr>
<tr>
<td>1119</td>
<td>016056</td>
<td>INCQ, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>1120</td>
<td>016057</td>
<td>MOVEM E, LINK (G)</td>
</tr>
<tr>
<td>1120</td>
<td>016058</td>
<td>MOVEV SLOT NO. TO LINK LIST</td>
</tr>
<tr>
<td>1121</td>
<td>016059</td>
<td>TTDCALL 3, CRLF</td>
</tr>
<tr>
<td>1122</td>
<td>01605A</td>
<td>JST GETCMD</td>
</tr>
<tr>
<td>1122</td>
<td>01605B</td>
<td>1GO ACCEPT NEXT COMMAND (SECTION B)</td>
</tr>
<tr>
<td>1123</td>
<td>01605C</td>
<td>TTDCALL 11, R</td>
</tr>
<tr>
<td>1124</td>
<td>01605D</td>
<td>TTDCALL 3, ASC1; /</td>
</tr>
<tr>
<td>1125</td>
<td>01605E</td>
<td>LINES MAY NOT BE INSERTED IN FRONT OF LINE ONE</td>
</tr>
<tr>
<td>1126</td>
<td>01605F</td>
<td>JST GETCMD</td>
</tr>
<tr>
<td>1126</td>
<td>016060</td>
<td>1GO ACCEPT NEXT COMMAND (SECTION B)</td>
</tr>
<tr>
<td>1127</td>
<td>016061</td>
<td>TTDCALL 11, R</td>
</tr>
<tr>
<td>1128</td>
<td>016062</td>
<td>BFULL21</td>
</tr>
<tr>
<td>1129</td>
<td>016063</td>
<td>TDCALL 3, ASC1; /</td>
</tr>
<tr>
<td>1130</td>
<td>016064</td>
<td>EDITING BUFFER ALREADY FULL</td>
</tr>
<tr>
<td>1131</td>
<td>016065</td>
<td>JST GETCMD</td>
</tr>
<tr>
<td>1131</td>
<td>016066</td>
<td>1GO ACCEPT NEXT COMMAND (SECTION B)</td>
</tr>
<tr>
<td>1132</td>
<td>016067</td>
<td>TTDCALL 11, R</td>
</tr>
<tr>
<td>1133</td>
<td>016068</td>
<td>BFULL21</td>
</tr>
<tr>
<td>1134</td>
<td>016069</td>
<td>MOVEM E, LINK (G)</td>
</tr>
<tr>
<td>1135</td>
<td>01606A</td>
<td>MOVEV SLOT NO. TO LINK LIST</td>
</tr>
<tr>
<td>1136</td>
<td>01606B</td>
<td>TTDCALL 3, ASC1; /</td>
</tr>
<tr>
<td>1137</td>
<td>01606C</td>
<td>EDITING BUFFER FULL</td>
</tr>
<tr>
<td>1138</td>
<td>01606D</td>
<td>JST GETCMD</td>
</tr>
<tr>
<td>1138</td>
<td>01606E</td>
<td>1GO ACCEPT NEXT COMMAND (SECTION B)</td>
</tr>
</tbody>
</table>
MOVE G,POINT2 IPUT BYTE POINTER TO SLOT ZERO IN AC-G
IDBP G ISKIP THE <35> AT BEGINNING OF LINE

2156.

1566.

1567.

1568.

1569.

1570.

1571.

1572.

1573.

1574.

1575.

1576.

1577.

1578.

1579.

1580.

1581.

1582.

1583.

1584.

1585.

1586.

1587.

1588.

1589.

1590.

1591.

1592.

1593.

1594.

1595.

1596.

1597.

1598.

1599.

1600.

1601.

1602.

1603.

1604.

1605.

1606.

1607.

1608.

1609.

1610.

1611.

1612.

1613.

1614.

1615.

1616.

1617.

1618.

1619.

1620.

1621.

1622.

1623.

1624.

1625.

1626.

1627.

1628.

1629.

1630.

1631.

1632.

1633.

1634.

1635.

1636.

1637.

1638.

1639.

1640.

1641.

1642.

1643.

1644.

1645.

1646.

1647.

1648.

1649.

1650.

1651.

1652.

1653.

1654.

1655.

1656.

1657.

1658.

1659.

1660.

1661.

1662.

1663.

1664.

1665.

1666.

1667.

1668.

1669.
SECTION L. INITIALIZE THE SPACE LIST BY PLACING THE NUMBERS 0 THROUGH 63 IN LOCATIONS SPACE THROUGH SPACE+63. THE SPACE LIST CONTAINS THE NUMBERS OF THE EMPTY SLOTS IN THE EDITING BUFFER.

SETBUF 11 HRKEI 1 SPACE+1 SETUP PUSHDOWN POINTER IN AC=J
MOVEL B.77
PUSH J.B.
PLACE CONTENTS OF AC=B IN THE SPACE LIST
JRST (3) RETURN


MOVE G,PC2 SAVE RETURN ADDRESS
MOVEL G,SETBUF INITIALIZE SPACE LIST (SECTION L)
SET28 P.B.
POP J.E.
MOVE J.E.1 REMOVE SLOT ZERO FROM SPACE LIST
MOVEL E.1 NEXT LINE WILL BE PLACED IN SLOT ONE
MOVE G,POINT2
MOVEL B,FRNUM HAS FRAME NUMBER BEEN READ?
JRNST FINPT NO, GO READ IT
MOVE A.36
IDPB A.G. PLACE <36> AT BEGINNING OF FIRST LINE IN SLOT ZERO
MOVE A,NXTFRM GET CURRENT FRAME NO. FROM NXTFRM
MOVEL A,FRNUM PRINT FRAME NUMBER IN FRNUM

PLACE FRAME NUMBER IN SLOT ZERO OF EDITING BUFFER
HRREI D,PDIST-1 PUT PUSHDOWN POINTER IN AC=O
IDI1 A.12 GET A DIGIT IN AC=O
IDIB A.68 CONVERT IT TO AN ASCII CHARACTER
PUSH D.B. PLACE CHARACTER IN PUSHDOWN LIST
JUMP A. +3 GET CHARACTER FROM PUSHDOWN LIST
IDPB A.G. PLACE CHARACTER IN THE EDITING BUFFER
IDIB A.G. GET NO. OF CHARACTERS REMAINING IN AC=O
IDPB A.G. PLACE SPACE AFTER FRAME NUMBER IN SLOT ZERO
<table>
<thead>
<tr>
<th>FILL</th>
<th>893</th>
<th>951</th>
<th>1042</th>
<th>1056</th>
<th>1096#</th>
<th>1098</th>
</tr>
</thead>
<tbody>
<tr>
<td>FINPT</td>
<td>1726</td>
<td>1737#</td>
<td>1738</td>
<td>1739#</td>
<td>1740#</td>
<td>1741#</td>
</tr>
<tr>
<td>FINPTZ</td>
<td>1726#</td>
<td>1735#</td>
<td>1736#</td>
<td>1737#</td>
<td>1738#</td>
<td>1739#</td>
</tr>
<tr>
<td>FLAGS</td>
<td>258#</td>
<td>263</td>
<td>267</td>
<td>266</td>
<td>265</td>
<td>264</td>
</tr>
<tr>
<td>FLAST</td>
<td>1763#</td>
<td>1766#</td>
<td>1767#</td>
<td>1768#</td>
<td>1769#</td>
<td>1770#</td>
</tr>
<tr>
<td>FHILL</td>
<td>1876#</td>
<td>1879#</td>
<td>1880#</td>
<td>1881#</td>
<td>1882#</td>
<td>1883#</td>
</tr>
<tr>
<td>FND</td>
<td>399#</td>
<td>399#</td>
<td>399#</td>
<td>399#</td>
<td>399#</td>
<td>399#</td>
</tr>
<tr>
<td>FNEND</td>
<td>572#</td>
<td>572#</td>
<td>572#</td>
<td>572#</td>
<td>572#</td>
<td>572#</td>
</tr>
<tr>
<td>FNDFH</td>
<td>838#</td>
<td>838#</td>
<td>838#</td>
<td>838#</td>
<td>838#</td>
<td>838#</td>
</tr>
<tr>
<td>FNDFH2</td>
<td>1872#</td>
<td>1875#</td>
<td>1878#</td>
<td>1881#</td>
<td>1884#</td>
<td>1887#</td>
</tr>
<tr>
<td>FNDFH3</td>
<td>1873#</td>
<td>1876#</td>
<td>1879#</td>
<td>1882#</td>
<td>1885#</td>
<td>1888#</td>
</tr>
<tr>
<td>FNDFH4</td>
<td>1874#</td>
<td>1877#</td>
<td>1880#</td>
<td>1883#</td>
<td>1886#</td>
<td>1889#</td>
</tr>
<tr>
<td>FNNO</td>
<td>614#</td>
<td>614#</td>
<td>614#</td>
<td>614#</td>
<td>614#</td>
<td>614#</td>
</tr>
<tr>
<td>FNTEO</td>
<td>1839#</td>
<td>1842#</td>
<td>1845#</td>
<td>1848#</td>
<td>1851#</td>
<td>1854#</td>
</tr>
<tr>
<td>FNUX</td>
<td>590#</td>
<td>590#</td>
<td>590#</td>
<td>590#</td>
<td>590#</td>
<td>590#</td>
</tr>
<tr>
<td>FNXTL</td>
<td>1730#</td>
<td>1733#</td>
<td>1736#</td>
<td>1739#</td>
<td>1742#</td>
<td>1745#</td>
</tr>
<tr>
<td>FRTINC</td>
<td>245#</td>
<td>248</td>
<td>251</td>
<td>254</td>
<td>257</td>
<td>260</td>
</tr>
<tr>
<td>FRTNUM</td>
<td>245#</td>
<td>248</td>
<td>251</td>
<td>254</td>
<td>257</td>
<td>260</td>
</tr>
<tr>
<td>FRMDY</td>
<td>193#</td>
<td>194</td>
<td>195</td>
<td>196</td>
<td>197</td>
<td>198</td>
</tr>
<tr>
<td>FJFTRM</td>
<td>193#</td>
<td>194</td>
<td>195</td>
<td>196</td>
<td>197</td>
<td>198</td>
</tr>
<tr>
<td>FSTLIN</td>
<td>1226#</td>
<td>1229#</td>
<td>1232#</td>
<td>1235#</td>
<td>1238#</td>
<td>1241#</td>
</tr>
<tr>
<td>FSTLIN</td>
<td>766#</td>
<td>776#</td>
<td>835</td>
<td>850</td>
<td>912</td>
<td>923</td>
</tr>
<tr>
<td>FUG</td>
<td>1793#</td>
<td>1796#</td>
<td>1800#</td>
<td>1803#</td>
<td>1806#</td>
<td>1809#</td>
</tr>
<tr>
<td>FUG</td>
<td>275#</td>
<td>278#</td>
<td>281</td>
<td>284</td>
<td>287</td>
<td>290</td>
</tr>
<tr>
<td>FUG</td>
<td>958#</td>
<td>1063#</td>
<td>1168#</td>
<td>1273#</td>
<td>1378#</td>
<td>1483#</td>
</tr>
<tr>
<td>GETHD</td>
<td>419#</td>
<td>422#</td>
<td>425#</td>
<td>428#</td>
<td>431#</td>
<td>434#</td>
</tr>
<tr>
<td>GETHD</td>
<td>459#</td>
<td>462#</td>
<td>465#</td>
<td>468#</td>
<td>471#</td>
<td>474#</td>
</tr>
<tr>
<td>GETHD</td>
<td>959#</td>
<td>962#</td>
<td>965#</td>
<td>968#</td>
<td>971#</td>
<td>974#</td>
</tr>
<tr>
<td>GETHD</td>
<td>1367#</td>
<td>1370#</td>
<td>1373#</td>
<td>1376#</td>
<td>1379#</td>
<td>1382#</td>
</tr>
<tr>
<td>GOH</td>
<td>143#</td>
<td>146#</td>
<td>149#</td>
<td>152#</td>
<td>155#</td>
<td>158#</td>
</tr>
<tr>
<td>GOH</td>
<td>259#</td>
<td>262#</td>
<td>265#</td>
<td>268#</td>
<td>271#</td>
<td>274#</td>
</tr>
<tr>
<td>HEL</td>
<td>455#</td>
<td>458#</td>
<td>461#</td>
<td>464#</td>
<td>467#</td>
<td>470#</td>
</tr>
<tr>
<td>HEL</td>
<td>465#</td>
<td>468#</td>
<td>471#</td>
<td>474#</td>
<td>477#</td>
<td>480#</td>
</tr>
<tr>
<td>HEL</td>
<td>164#</td>
<td>167#</td>
<td>170#</td>
<td>173#</td>
<td>176#</td>
<td>179#</td>
</tr>
<tr>
<td>IFL</td>
<td>164#</td>
<td>167#</td>
<td>170#</td>
<td>173#</td>
<td>176#</td>
<td>179#</td>
</tr>
<tr>
<td>IFL</td>
<td>164#</td>
<td>167#</td>
<td>170#</td>
<td>173#</td>
<td>176#</td>
<td>179#</td>
</tr>
<tr>
<td>IFL</td>
<td>164#</td>
<td>167#</td>
<td>170#</td>
<td>173#</td>
<td>176#</td>
<td>179#</td>
</tr>
</tbody>
</table>
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

III = FROM PRL.MAC AND WRTEO.MAC, LOCATED IN SECTION H

IIT = FROM WRTEO.MAC, LOCATED IN SECTION H

INCHR = FROM PRL.MAC, WRTEO.MAC, AND AUX2.MAC; LOCATED IN
SECTION B

IID = FROM PRL.MAC AND WRTEO.MAC, LOCATED IN SECTION E

IIOB = FROM PRL.MAC, LOCATED IN SECTION F

IOT = FROM PRL.MAC AND WRTEO.MAC, LOCATED IN SECTION G
Macro 47(204)-7 08134 2-Aug-74 Page 1-1

IOUTCHR - FROM PERI, MAC, WRTED, MAC, AND AUX2, MAC, LOCATED IN SECTION C

IOUTSTR - FROM PERI, MAC, LOCATED IN SECTION D

EXTERNAL MEMORY REFERENCES

IBUF - BUFFER HEADER FOR DISK INPUT (3 WORDS)

OBUF - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)

PC2 - RETURN ADDRESS

PDST - PUSH DOWN LIST FOR STORING ASCII DIGITS OF A NUMBER

BEING CONVERTED FROM BINARY TO ASCII CODE (8 WORDS)

%000000

ENTRY INCH, OUTCHR, OUTSTR, IOC, IOT, IOLN

INTERN IIT, IIT2, IOL, IOC

EXTERN IBUF, OBUF, PC2, PDST

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 CHANNEL 1, THE CHARACTER IS PLACED IN AC-A,

INCHR SLOSE IBUF+2

INCR = ASCE IBUF+2

DECREMENT CHARACTER COUNT

JST GETBUF

IF BUFFER IS EMPTY, GO GET ANOTHER BUFFER

GETNXI ILDB A, IBUF+1

PUT CHARACTER IN AC-A

JST (Q)

RETURN

GETBUF IN 1

FILL BUFFER

JST INCHR

IF NO ERROR, GO GET NEXT CHARACTER

STATE 1,20000

END OF FILE?
REPRODUCED WITH PERMISSION OF THE COPYRIGHT OWNER. FURTHER REPRODUCTION PROHIBITED WITHOUT PERMISSION.

SECTION C: THIS SUBROUTINE OUTPUTS A CHARACTER TO THE DISK ON CHANNEL 1. THE CHARACTER MUST BE IN AC-A.

SECTION 0: THIS SUBROUTINE OUTPUTS A STRING OF CHARACTERS TO THE DISK. THE CHARACTERS MUST BE STORED IN ASCII FORMAT IMMEDIATELY FOLLOWING THE CALLING INSTRUCTION.

SECTION E: THIS SUBROUTINE OUTPUTS AN INTEGER TO THE DISK. THE INTEGER MUST BE IN AC-A.

107 000034 254 00 0 17 000000 JST (O) IYES, RETURN
108 TCALL 3:CASC12 /
109 DISK INPUT ERROR ON CHANNEL 1 = AUX2
110 000034 051 03 0 00 000024 I] HALT
111 000035 254 04 0 00 000000

SECTION D: THIS SUBROUTINE OUTPUTS A STRING OF CHARACTERS TO THE DISK. THE CHARACTERS MUST BE STORED IN ASCII FORMAT IMMEDIATELY FOLLOWING THE CALLING INSTRUCTION.

112 000036 375 00 0 00 000002 OUTCHR: INDEX OF BUFFER ++1 DECREMENT CHARACTER COUNT
113 000037 254 00 0 00 000022 JST OUTBUF: IF BUFFER FULL, GO EMPTY IT
114 000038 051 03 0 00 000001 PUTFXT: IDNO A,OBUF+1 PUT CHARACTER IN OUTPUT BUFFER
115 000039 254 00 0 17 000000 JST (O) IRETURN
116 00003A 057 02 0 00 000000 PUTFUI: OUT 2: EMPTY BUFFER
117 00003B 254 00 0 00 000016 JST OUTCHR: IF NO ERROR, GO PUT CHARACTER IN BUFFER
118 00003C 051 03 0 00 000155 TCALL 3:CASC12 /
119 DISK OUTPUT ERROR ON CHANNEL 2 = AUX2
120 00003D 254 04 0 00 000000 I] HALT
121 00003E

SECTION E: THIS SUBROUTINE OUTPUTS AN INTEGER TO THE DISK. THE INTEGER MUST BE IN AC-A.

122 00003F 252 17 0 00 000003 IDIO MOVE Q,PCC1 SAVE RETURN ADDRESS
123 000040 551 04 0 00 777777 MRZ: D,POLST+1 SET UP PUSH DOWN POINTER IN AC-0
124 000041 231 01 0 00 000122 IDIVI A,12 GET DIGIT IN AC-A
125 000042 435 02 0 00 000050 IDRI B,30 CONVERT TO ASCII CODE
126 000043 281 04 0 00 000020 PUSH D,B PUT CHARACTER IN PUSH DOWN LIST
127 000044 326 01 0 00 000026 JUMP A,100+2 100 GET NEXT DIGIT IF THERE IS ONE
128 000045 262 04 0 00 000020 JOP D,B GET CHARACTER FROM PUSH DOWN LIST
129 000046 265 17 0 00 000016 JSP Q,OUTCHR OUTPUT CHARACTER TO DISK...
130 000047 683 04 0 00 777777 TLNE D,777777 ANY CHARACTERS LEFT?
131 000048 254 00 0 00 000041 JST +3 IYES, GO GET NEXT CHARACTER
132 000049 254 01 0 00 000034 JST @PCC1 OTHERWISE, RETURN
SECTION F. THIS SUBROUTINE OUTPUTS AN OCTAL NUMBER TO THE
IDISK. THE NUMBER MUST BE IN AC-A.

160 200257 202 17 0 00 000061 10081 MOVEB Q,PC2 1SAVE RETURN ADDRESS
161 200256 202 17 0 00 000100 HRRZI D,DPLST-1 1SET UP PUSH DOWN POINTER IN AC-D
162 200255 202 17 0 00 000101 IDIVI A,10 1GET DIGIT IN AC-A
163 200254 202 17 0 00 000100 IDVRI A,60 1CONVERT TO ASCII CODE
164 200253 202 17 0 00 000100 PUSB D,B 1PUT CHARACTER IN PUSH DOWN LIST
165 200252 202 17 0 00 000100 JUMPN A,1006-2 1GO GET NEXT DIGIT IF THERE IS ONE
166 200251 202 17 0 00 000100 POP D,A 1GET CHARACTER FROM PUSH DOWN LIST
167 200250 202 17 0 00 000100 JSP B,OUTCHR 1OUTPUT CHARACTER TO DISK
168 200249 202 17 0 00 000100 TLSB D,77777 1ANY CHARACTERS LEFT?
169 200248 254 00 0 00 000591 JRS T,-3 1YES, GO GET NEXT CHARACTER
170 200247 254 00 0 00 000621 JRS T,*PC1 1OTHERWISE, RETURN

SECTION G. THIS SUBROUTINE OUTPUTS AN INTEGER TO THE USER'S
TERMINAL. THE INTEGER MUST BE IN AC-A.

171 200242 202 17 0 01 000061 10101 MOVEB Q,PC2 1SAVE RETURN ADDRESS
172 200241 202 17 0 01 000300 HRRZI D,DPLST-1 1SET UP PUSH DOWN POINTER IN AC-D
173 200240 202 17 0 01 000100 IDIVI A,12 1GET DIGIT IN AC-A
174 200239 202 17 0 01 000100 IDVRI A,60 1CONVERT TO ASCII CODE
175 200238 202 17 0 01 000100 PUSB D,B 1PUT CHARACTER IN PUSH DOWN LIST
176 200237 202 17 0 01 000100 JUMPN A,1010-2 1GO GET NEXT DIGIT IF THERE IS ONE
177 200236 202 17 0 01 000100 POP D,A 1GET CHARACTER FROM PUSH DOWN LIST
178 200235 202 17 0 01 000100 TDCALL 11,A 1TYPE THE CHARACTER
179 200234 202 17 0 01 000100 TLSB D,77777 1ANY CHARACTERS LEFT?
180 200233 254 00 0 00 000790 JRS T,-3 1YES, GO GET NEXT CHARACTER
181 200232 254 00 0 00 000621 JRS T,*PC1 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
I1IT ENTRY POINT IS USED, LEADING SPACES ARE IGNORED.

182 200075 400 02 0 00 000200 I1T1 SETE B, 1INPUT ASCII CHARACTER
183 200074 400 02 0 00 000201 TTCALL A,4,A 1INPUT ASCII CHARACTER
184 200073 366 01 0 00 000400 CALL A,40 1IS CHARACTER A SPACE?
185 200072 353 00 0 00 000040 JRS T,,-2 1YES, GO INPUT NEXT CHARACTER
186 200071 254 03 0 00 001031 JRS T,,-3 1NO
187 200070 254 03 0 00 000031 I1T11 TTCALL A,4,A 1INPUT NEXT CHARACTER
188 200069 301 01 0 00 000660 CALL A,60 1RETURN IF CHARACTER IN NOT A DIGIT
189 200068 300 01 0 00 000060 CALL A,60 1MULTIPLY BY 10
190 200067 271 02 0 00 77777 ADDI B,-80(A) 1ADD VALUE OF DIGIT TO NO. IN AC-B
191 200066 254 00 0 00 001201 JRS T,I1T1 1GO GET NEXT DIGIT
192 200065 400 02 0 00 000000 I1T21 SETE B, 1INPUT ASCII CHARACTER

(Continues on next page)
SECTION I. THIS SUBROUTINE INPUTS AN INTEGER FROM THE DISK AND PLACES IT IN AC-B.

000112 254 00 00 000103  JST 113
000113 262 17 00 000016 11D1  MOVEH Q,16  ISTORE RETURN ADDRESS IN AC-16
000114 400 02 00 000000 3132  SETE B,
000115 265 17 00 000041 11D1  JSP Q,INCHR  INPUT CHARACTER
000116 301 01 00 000060 3132  CAIL A,60
000117 303 01 00 000071 3132  CAILE A,71
000120 254 00 00 000000 11D1  IJRT (16)  JRETURN IF CHARACTER IS NOT A DIGIT
000121 221 02 00 000012 3132  IMULI B,12  IMULTIPLY BY 10
000122 271 02 01 777720 3132  ADDI B,-60(A)  ADD VALUE OF DIGIT TO NO. IN AC-B
000123 254 00 00 000151 11D1  IJRT (101)  IG0 GET NEXT DIGIT

NO ERRORS DETECTED

PROGRAM BREAK IS 000146

5K CORE USED
TITLE LOTLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 1-AUG-73

PART OF LEDIT.SHR, VERSION 3

LOTLOW.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,FROMINC,INBUF,BUFFER,LINBUF,OLDSTR,TRMLK,JOBNO,
INTERN FRMNUM,FRMTP,LINK,SPACE,FSTFRM,LSTFRM,LSTLIN,LSTLIN,
INTERN STORE,INAH,DAH,PC1,IBUF,OBUF,PC2,POLST,PPN,PPD,CUTNUM,PO4

INTERN INBUF,OUTBP2,OUTBUF

000137  JOBVER+137
000337  LOC JOBVER
000301  AND 381,1
000000  RLOCG 0

000301
000301  ARGMT
000301  JNNO,1  BLOCK 1  JOB NUMBER
000301  TRMLK  BLOCK 3  TRAMP, ARGUMENTS
000301  INTBLK1 BLOCK 5  INTERRUPT BLOCK IS STORED HERE
000321  INTLOC1 BLOCK 12  INTERRUPT ROUTINE IS STORED HERE
000301  IBUF1 BLOCK 3  USED BY WRTE5,MAC AND AUX2,MAC
000301  OBUF1 BLOCK 3  USED BY WRTE5,MAC AND AUX2,MAC
000301  INAH1 BLOCK 4  USED BY PERL,MAC AND WRTE5,MAC
000301  ONAH1 BLOCK 4  USED BY PERL,MAC, LPRINT,MAC, AND WRTE5,MAC
000250  PPN1 BLOCK 1  USED BY PERL,MAC
000251  INBUF1 BLOCK 283  USED BY PERL,MAC AND WRTE5,MAC
000294  OUTBUF1 BLOCK 283  USED BY WRTE5,MAC

ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN Lragment, MAC,
IRPTLLOW.MAC, AND FRMLLOW.MAC.

PC11  BLOCK 1  USED BY PERL,MAC AND WRTE5,MAC
PC21  BLOCK 1  USED BY WRTE5,MAC AND AUX2,MAC
PC31  BLOCK 1  USED BY WRTE5,MAC
PC41  BLOCK 1  USED BY WRTE5,MAC
POLS1 BLOCK 10  USED BY WRTE5,MAC AND AUX2,MAC
FRMLNC1 BLOCK 1  USED BY WRTE5,MAC
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>003474</td>
</tr>
<tr>
<td>55</td>
<td>003475</td>
</tr>
<tr>
<td>56</td>
<td>003476</td>
</tr>
<tr>
<td>57</td>
<td>003477</td>
</tr>
<tr>
<td>58</td>
<td>003478</td>
</tr>
<tr>
<td>59</td>
<td>003479</td>
</tr>
<tr>
<td>60</td>
<td>003502</td>
</tr>
<tr>
<td>61</td>
<td>003503</td>
</tr>
<tr>
<td>62</td>
<td>003504</td>
</tr>
<tr>
<td>63</td>
<td>003505</td>
</tr>
<tr>
<td>64</td>
<td>003506</td>
</tr>
<tr>
<td>65</td>
<td>003507</td>
</tr>
<tr>
<td>66</td>
<td>003508</td>
</tr>
<tr>
<td>67</td>
<td>003509</td>
</tr>
<tr>
<td>68</td>
<td>003510</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
TITLE LRPT

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT COMPLETED 16-JUL-73

PART OF LRPT, SHR, VERSION 3

SPECIAL NOTE!

LRPT, MAC CONTAINS THE ONLY ENTRY POINT TO THE LRPT HIGH SEGMENT.
THIS ENTRY POINT IS LOCATED IN SECTION B OF THIS PROGRAM.

THE LRPT HIGH SEGMENT IS COMPOSED OF THE FOLLOWING SUBPROGRAMS:

1. LRPT, MAC
2. RPRT, MAC
3. LPRT, MAC
4. RPRT, MAC
5. AUX, MAC
6. RPLMT, 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 LRPT
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SAVE MONITOR COMMAND.

THE LRPT HIGH SEGMENT EXECUTES THE FOLLOWING LEARN SYSTEM COMMANDS:

1. BYE
2. CLEAR
3. DELETE
4. KJOB
5. MONITOR
6. REPORT
7. UPDATE

LRPT, MAC INTERPRETS LEARN SYSTEM COMMANDS AND CALLS THE PROPER SUB-
PROGRAMS TO EXECUTE THEM. IF A COMMAND CANNOT BE EXECUTED BY THE
LRPT HIGH SEGMENT, THEN LRPT, MAC PLACES A COMMAND INDEX IN LOCATION
STORE AND REPLACES ITSELF WITH THE HIGH SEGMENT WHICH CAN EXECUTE THE
COMMAND. WHEN LRPT, 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 LRPT, MAC
SECTION A. PERFORMS THE GETSEG UNO.

SECTION B. CONTAINS THE ENTRY POINT TO THE LRPT 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 LPORS HIGH SEGMENT.

SECTION I. CALLS THE PROPER LRPT SUBPROGRAM TO EXECUTE A LEARN COMMAND.

SECTION J. EXECUTES A "BYE" COMMAND.

SECTION K. EXECUTES A "KJOB" COMMAND.

ENTRY POINT TO LRPT.MAC

RETURN - FROM ANOTHER HIGH SEGMENT, LOCATED IN SECTION B

EXTERNAL SUBROUTINES CALLED BY LRPT.MAC

CLR - EXECUTES A "CLEAR" COMMAND, LOCATED IN RPRT.MAC (SECTION D)

DEL - EXECUTES A "DELETE" COMMAND, LOCATED IN RPRT.MAC (SECTION A)

EOILIN - 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 RPTLOW.MAC)

ARGMT = ARGUMENTS FOR GETSEG UNO (6 WORDS)

STORE = COMMAND INDEX IS SAVED HERE DURING A GETSEG

TRMBLK = TRHOP, ARGUMENTS (3 WORDS)
SECT A. GET THE PROPER HIGH SEGMENT TO EXECUTE THE COMMAND WHICH WAS TYPED BY THE USER. EXECUTION RESUMES IN "LEARN.MAC" ILEDIT.MAC. OR "LPROG.MAC."

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

GETSEG: GETSEG A.

HALT

SECT 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.

RETURNI MOVE B,STORE "GET THE COMMAND INDEX IN AC-B"

JST R.(B)

AB 0,REPORT ISEE SECTION I

AB 0,UPDATE ISEE SECTION I

AB 0,CLEAR ISEE SECTION I

AB 0,DELETE ISEE SECTION I

SECT C. INPUT THE FIRST WORD IN A LEARN SYSTEM COMMAND.

CMDI MOVE A,TRNBLK "MOVE ADDRESS OF TRNBLK, ARGUMENTS TO AC-A"

TRNBLK A;

JSKIP IF TTY OUTPUT BUFFER IS NOT EMPTY

JST EMPTY

MOVE I,1

SLEEP B;

ISLEEP A SEC BEFORE CHECKING BUFFER AGAIN

EMPTY: TDCALL 14 IGO CHECK BUFFER AGAIN

JFLC

TDCALL 3,ASCIE I

ENTER COMMAND
II

J'!

- z_

~ ~

ooz

wo

~0

0

\[\text{SECTION G, STORE COMMAND INDEX AND GETSEG ARGUMENTS BEFORE GETTING THE EDIT HIGH SEGMENT.}\]

\[\text{SECTION H, STORE COMMAND INDEX AND GETSEG ARGUMENTS BEFORE GETTING THE LPROS HIGH SEGMENT.}\]

\[\text{SECTION I, JUMP TO THE PROPER ROUTINE TO EXECUTE THE COMMAND WHICH WAS TYPED BY THE USER.}\]
SECTION J. EXECUTE THE BYE COMMAND BY RUNNING THE LOGOUT PROGRAM.

RUNARG: SIXBIT /SYS/

SECTION K. EXECUTE THE KJOB COMMAND BY RUNNING THE KJOB PROGRAM.

RARG: SIXBIT /SYS/

NO ERRORS DETECTED

PROGRAM BREAK IS D33388

3K CORE USED
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-PROG 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 0-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 7-131 RIGHT-WRONG-NEUTRAL INDICATOR
               BITS 14-231 ANSWERING TIME IN SECONDS
               BIT 35 IS SET IF CALC MODE WAS USED

FOR OTHER FRAME TYPES, THE 1ST WORD IS THE SAME AS FOR 0-FRAMES, AND THE 2ND WORD IS ZERO.

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.

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 <13> ONLY.

ENTRY POINTS TO RPRT.MAC

CLR = FROM LRPRTR, MAC (SECTION 1), LOCATED IN SECTION H
DEL = FROM LRPRTR, MAC (SECTION 1), LOCATED IN SECTION A
RPRT = FROM LRPRTR, MAC (SECTION 1), LOCATED IN SECTION A
UPDT = FROM LRPRTR, MAC (SECTION 1), LOCATED IN SECTION I

EXTERNAL SUBROUTINES CALLED BY RPRT.MAC

I1 - INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B;
     LOCATED IN AUX3.MAC (SECTION C)
I12 - INPUTS AN INTEGER FROM A USER'S TERMINAL INTO AC-B
107 1ST DIGIT IS ALREADY IN AC-A, LOCATED IN AUX3,MAC (SECTION C)
108
109
110 INCHR = INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN AUX3,MAC (SECTION B)
111
112
113 INWRT = INPUTS A WORD OF DATA FROM THE DISK INTO AC-A, LOCATED IN AUX3,MAC (SECTION G)
114
115
116 IORD = OUTPUTS THE INTEGRAL IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION D)
117
118
119 LPTRPT = GENERATES DETAILED REPORTS, LOCATED IN LPTRPT,MAC (SECTION A)
120
121
122 OTOA = OUTPUTS THE DATE IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION J)
123
124
125 OTTIM2 = OUTPUTS THE TIME IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION K)
126
127
128 OUTCHR = OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION F)
129
130
131 OUTDAT = OUTPUTS THE CURRENT DATE TO THE DISK, LOCATED IN AUX3,MAC (SECTION J)
132
133
134 OUTLES = OUTPUTS THE LESSON NAME IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION I)
135
136
137 OUTPPN = OUTPUTS THE PROGRAMMER NO. IN AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION L)
138
139
140 OUTSTR = OUTPUTS A STRING OF CHARACTERS TO THE DISK, LOCATED IN AUX3,MAC (SECTION E)
141
142
143 OUTTIM = OUTPUTS THE CURRENT TIME TO THE DISK, LOCATED IN AUX3,MAC (SECTION K)
144
145
146 OUTTM = OUTPUTS LESSON TIME OR ANSWERING TIME FROM AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION M)
147
148
149 OUTWRT = OUTPUTS A WORD OF DATA FROM AC-A TO THE DISK, LOCATED IN AUX3,MAC (SECTION W)
150
151
152 RTJUST = RIGHT JUSTIFIES THE NUMBER IN AC-A, LOCATED IN AUX3,MAC (SECTION N)
153
154
155 RPRINT = PLACES THE REPORT FILE IN PRINT QUEUE, LOCATED IN RPRINT,MAC (SECTION A)
156
157
158 EXTERNAL MEMORY REFERENCES (ALL IN RPTLOW,MAC)
160  ABO - REPORT COMMAND SWITCH
161
162  ANSTOP - TOTAL ANSWERING TIME IS ACCUMULATED HERE
163
164  DATA - LIST OF SUMMARY DATA (16 WORDS)
165
166  DAYT - DATES SPECIFIED IN REPORT COMMAND (2 WORDS)
167
168  DEV - DEVICE SPECIFIED IN REPORT COMMAND
169
170  EINST - NO. OF TIMES LESSON WAS EXECUTED
171
172  FRAME - FRAME NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)
173
174  IBUF - DISK BUFFER HEADER (3 WORDS)
175
176  IUFF - DISK INPUT BUFFER (131 WORDS)
177
178  IUFF2 - DISK BUFFER HEADER (3 WORDS)
179
180  INAM - LOOKUP ARGUMENTS (4 WORDS)
181
182  INAM2 - LOOKUP ARGUMENTS (4 WORDS)
183
184  LESSON - LESSON NAME SPECIFIED IN REPORT COMMAND
185
186  LIM1 - USED TO STORE INDEX BY QUICKSORT
187
188  LIM2 - USED TO STORE INDEX BY QUICKSORT
189
190  MONTH - MONTH SPECIFIED IN REPORT OR DELETE COMMAND
191
192  NTSTOP - NO. OF NEUTRAL RESPONSES IS ACCUMULATED HERE
193
194  OBUFF - DISK BUFFER HEADER (3 WORDS)
195
196  OBUFF - DISK OUTPUT BUFFER (131 WORDS)
197
198  OBUF2 - DISK BUFFER HEADER (3 WORDS)
199
200  ONAM - ENTER ARGUMENTS (4 WORDS)
201
202  ONAM2 - ENTER ARGUMENTS (4 WORDS)
203
204  PARAM - BEGINNING OF FRAME DATA LIST
205
206  PC1 - RETURN ADDRESS IS STORED HERE
207
208  PPNI1 - PROGRAMMER NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)
209
210  RESTOT - TOTAL NO. OF RESPONSES IS ACCUMULATED HERE
211
212
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>671</td>
<td>02643</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>672</td>
<td>02644</td>
<td>IOUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>673</td>
<td>02645</td>
<td>MOVE A,DAYS+1</td>
</tr>
<tr>
<td>674</td>
<td>02646</td>
<td>IGET THE 2ND DATE</td>
</tr>
<tr>
<td>675</td>
<td>02647</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>676</td>
<td>02648</td>
<td>IOUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>677</td>
<td>02649</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>678</td>
<td>02650</td>
<td>MOVE A,PPNL</td>
</tr>
<tr>
<td>679</td>
<td>02651</td>
<td>IGET THE 1ST PROGRAMMER NO.</td>
</tr>
<tr>
<td>680</td>
<td>02652</td>
<td>JSP Q,OUTPPNL</td>
</tr>
<tr>
<td>681</td>
<td>02653</td>
<td>IOUTPUT THE NO.</td>
</tr>
<tr>
<td>682</td>
<td>02654</td>
<td>MOVE A, &quot;\n&quot;</td>
</tr>
<tr>
<td>683</td>
<td>02655</td>
<td>JSP Q,OUTCHR</td>
</tr>
<tr>
<td>684</td>
<td>02656</td>
<td>IGET THE 2ND PROGRAMMER NO.</td>
</tr>
<tr>
<td>685</td>
<td>02657</td>
<td>JSP Q,OUTPPNL+1</td>
</tr>
<tr>
<td>686</td>
<td>02658</td>
<td>IOUTPUT THE NO.</td>
</tr>
<tr>
<td>687</td>
<td>02659</td>
<td>MOVE A,PPNL</td>
</tr>
<tr>
<td>688</td>
<td>02660</td>
<td>IGET THE PROGRAMMER NO. FOR THIS REPORT</td>
</tr>
<tr>
<td>689</td>
<td>02661</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>690</td>
<td>02662</td>
<td>IOUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>691</td>
<td>02663</td>
<td>ASCIZ /</td>
</tr>
</tbody>
</table>

**Student Date**

<table>
<thead>
<tr>
<th>Time</th>
<th>Right</th>
<th>Wrong</th>
<th>Neut</th>
<th>Time Up</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Lesson Date**

<table>
<thead>
<tr>
<th>Time</th>
<th>Right</th>
<th>Wrong</th>
<th>Neut</th>
<th>Time Up</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MOVEI A,40
JSP Q,OUTCHR
OUTPUT SPACE
JSP Q,OUTCHR
IOUTPUT THE STUDENT'S PROGRAMMER NO.
JSP Q,TJUJS
RIGHT JUSTIFY IT
JSP Q,OUTCHR
OUTPUT THE PROGRAMMER NO.
MOVEI A,11
JSP Q,OUTCHR
OUTPUT A TAB
JSP Q,OUTCHR
IGET THE LESSON EXECUTION DATE
JSP Q,ODAT2
OUTPUT THE DATE
JSP Q,OUTSTR
OUTPUT THE FOLLOWING STRING
ASCII /

MOVE A,DATA+4
IGET THE TOTAL ELAPSED TIME FOR THE LESSON
ADD A,DATA+2
IGET THE TOTAL ELAPSED TIMES FROM OTHER LESSONS
MOVEI A,11
IGET AQD TO 1 FOR OUTM SUBROUTINE
JSP Q,OUTCHR
OUTPUT THE ELAPSED LESSON TIME
MOVEI A,11
JSP Q,OUTCHR
OUTPUT A TAB
JSP Q,OUTCHR
OUTPUT THE FOLLOWING STRING
ASCII /

MOVE A,DATA+11
IGET THE TOTAL ANSWERING TIME FOR THE LESSON
ADD A,ANSTOT
IASS IT TO THE ANSWERING TIMES FROM OTHER LESSONS
DIV A,1000
IDIVIDE TO GET THE TIME IN MINUTES
CALL G,103
AD A,1000 IF REMAINDER > 30, ROUND UP TO NEXT MINUTE
JSP Q,RTUJS
RIGHT JUSTIFY NO.
JSP Q,IOD
OUTPUT THE ANSWERING TIME
MOVEI A,11
JSP Q,OUTCHR
OUTPUT A TAB
JSP Q,OUTSTR
OUTPUT THE FOLLOWING STRING
ASCII /

MOVE A,DATA+5
IGET THE TOTAL NO. OF CORRECT ANSWERS
ADD A,RTTOT
IADD IT TO THE NO. OF CORRECT ANSWERS FROM OTHER LESSONS
MOV A,RESTOT
IADD IT TO THE TOTAL RESPONSES FOR ALL LESSONS
MOV C A
ISAVE THE NUMBER IN AC
MOVEI A,40
JSP Q,OUTCHR
OUTPUT A SPACE
MOVE A,C
INPUT NO. BACK IN AC-A
JSP Q,TJUJS
RIGHT JUSTIFY NO.
JSP Q,IOD
OUTPUT NO. OF CORRECT ANSWERS
JSP Q,OUTSTR
OUTPUT THE FOLLOWING STRING
ASCII /

MOVE A,DATA+6
IGET THE TOTAL NO. OF WRONG ANSWERS
ADD A,ANSTOT
IADD IT TO NO. OF WRONG ANSWERS FROM OTHER LESSONS
ADD A,RESTOT
IADD IT TO THE TOTAL RESPONSES FOR ALL LESSONS
MOV C A
ISAVE THE NO. IN AC-C
MOVEI A,40
JSP Q,OUTCHR
OUTPUT A SPACE
MOVE A,C
INPUT NO. BACK IN AC-A
JSP Q,TJUJS
RIGHT JUSTIFY NO.
JSP Q,IOD
OUTPUT THE TOTAL NO. OF WRONG ANSWERS
MOVEI A,11
JSP Q,OUTCHR
OUTPUT A TAB
MOV A,DATA+7
IGET THE TOTAL NO. OF NEUTRAL RESPONSES
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>847</td>
<td>LESFDQ STR G,OUTSTR ,OUTPUT THE FOLLOWING STRING ASCII / AVE. ELAPSED TIME /</td>
</tr>
<tr>
<td>848</td>
<td>MOVE A,TINST I GET THE TOTAL ELAPSED TIME FOR ALL LESSONS</td>
</tr>
<tr>
<td>849</td>
<td>IDIV A,EXITOT I DIVIDE BY NO. OF TIMES LESSON WAS EXECUTED</td>
</tr>
<tr>
<td>850</td>
<td>JSP G,OUTTM I OUTPUT THE AVERAGE ELAPSED TIME</td>
</tr>
<tr>
<td>851</td>
<td>JSP G,OUTSTR I OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>852</td>
<td>MOVE A,ANSTOT I GET THE TOTAL ANSWERING TIME FOR ALL LESSONS</td>
</tr>
<tr>
<td>853</td>
<td>IDIV A,EXITOT I DIVIDE BY NO. OF TIMES THE LESSON WAS EXECUTED</td>
</tr>
<tr>
<td>854</td>
<td>JSP G,OUTTM I OUTPUT THE AVERAGE ANSWERING TIME</td>
</tr>
<tr>
<td>855</td>
<td>JSP G,OUTSTR I OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>856</td>
<td>ASCII / MINUTES</td>
</tr>
<tr>
<td>857</td>
<td>MOVE A,RTNINST I GET THE TOTAL CPU TIME FOR ALL LESSONS</td>
</tr>
<tr>
<td>858</td>
<td>IDIV A,EXITOT I DIVIDE BY NO. OF TIMES LESSON WAS EXECUTED</td>
</tr>
<tr>
<td>859</td>
<td>JSP G,OUTTM I OUTPUT THE SECONDS OF CPU TIME</td>
</tr>
<tr>
<td>860</td>
<td>JSP G,OUTSTR I OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>861</td>
<td>ASCII / SECONDS</td>
</tr>
<tr>
<td>862</td>
<td>TOTAL NO. OF RESPONSES /</td>
</tr>
<tr>
<td>863</td>
<td>MOVE A,RESETST I GET THE TOTAL NO. OF RESPONSES FOR ALL LESSONS</td>
</tr>
<tr>
<td>864</td>
<td>JSP G,IOO D OUTPUT THE TOTAL NO. OF RESPONSES</td>
</tr>
<tr>
<td>Line No.</td>
<td>Instruction</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>881</td>
<td>JSP Q;OUTSTR</td>
</tr>
<tr>
<td>882</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>883</td>
<td>TOTAL NO. OF CORRECT RESPONSES /</td>
</tr>
<tr>
<td>884</td>
<td>MOVE A;RTTOT</td>
</tr>
<tr>
<td>885</td>
<td>JSP Q;10D</td>
</tr>
<tr>
<td>886</td>
<td>JSP Q;OUTSTR</td>
</tr>
<tr>
<td>887</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>888</td>
<td>TOTAL NO. OF WRONG RESPONSES /</td>
</tr>
<tr>
<td>889</td>
<td>MOVE A;HTTOT</td>
</tr>
<tr>
<td>890</td>
<td>JSP Q;10D</td>
</tr>
<tr>
<td>891</td>
<td>JSP Q;OUTSTR</td>
</tr>
<tr>
<td>892</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>893</td>
<td>TOTAL NO. NEUTRAL RESPONSES /</td>
</tr>
<tr>
<td>894</td>
<td>MOVE A;NTTOT</td>
</tr>
<tr>
<td>895</td>
<td>JSP Q;10D</td>
</tr>
<tr>
<td>896</td>
<td>JSP Q;OUTSTR</td>
</tr>
<tr>
<td>897</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>898</td>
<td>TOTAL NO. OF TIME UPS /</td>
</tr>
<tr>
<td>899</td>
<td>MOVE A;TUTTOT</td>
</tr>
<tr>
<td>900</td>
<td>JSP Q;10D</td>
</tr>
<tr>
<td>901</td>
<td>JSP Q;OUTSTR</td>
</tr>
<tr>
<td>902</td>
<td>ASCIZ /</td>
</tr>
<tr>
<td>903</td>
<td>ERROR RATE /</td>
</tr>
<tr>
<td>904</td>
<td>MOVE B;RTTOT</td>
</tr>
<tr>
<td>905</td>
<td>ADD B;WGTOT</td>
</tr>
<tr>
<td>906</td>
<td>MOVE A;WGTOT</td>
</tr>
<tr>
<td>907</td>
<td>IMULI A;490880</td>
</tr>
<tr>
<td>908</td>
<td>IDIV A;B</td>
</tr>
</tbody>
</table>
IDIVI A, +D10
CALL B, 9

BSR A

IDIVI A, +D10
BSR A

MOVEM B, +D10
MOVEM B, WOTOT

JSP Q, IOD

MOVEM A, ".

JSP Q, OUTCHR

MOVEM A, WOTOT

GET THE FRACTIONAL PART

JSP Q, IOD

OUTPUT FRACTIONAL PART

JSP Q, OUTSTR

OUTPUT THE FOLLOWING STRING

ASCII /

/RESET CLOSED

CLOSE 2,

CLOSE 1,

SETSTS 1,0

RESTORE ASCII MODE

MOVEM A, [70000000]

MOVEM A, IBUF+1

RESTORE ASCII BYTE POINTER

MOVEM B, 'Y'

MOVEM B, 'T'

CAMM B, DEV

IS OUTPUT DEVICE TTY?

JAST TYPFIL

YES, GO TO TYPFIL

JSP Q, PRINT

ELSE, PLACE REPORT FILE IN PRINT QUEUE

TCALL 3, IASCIZ /

REPORT FILE IN PRINT QUEUE

JAST QPC1

RETURN TO LRPRT, MAC

TYPFIL

MOVEM B, ONAM I GET THE REPORT FILE NAME

MOVEM B, INAM I PUT IT IN INAM

MOVEM B, LIST+1

MOVEM B, INAM+1 I PUT LIST IN INAM+1

SETEM INAM+3

LOOKUP 1, INAM I LOOKUP REPORT FILE

WALT

JSP Q, INCHR

INPUT CHARACTER FROM REPORT FILE

STAT 1, 20000

END OF FILE?

JAST +3

YES, JUMP

TDCALL 1, A

ELSE TYPE CHARACTER

JAST +4

GO INPUT NEXT CHARACTER

SETEM INAM

SETEM INAM+3

RENAM E 1, INAM

DELETE REPORT FILE

WALT

JAST QPC1

RETURN TO LRPRT, MAC

/SECTI ON E: PRODUCE A STUDENT REPORT (SUMMARY FORM)

STDRPI SETEM STOTSET SET NO. OF STUDENTS EXECUTING LESSONS TO ZERO

JSP Q, OUTSTR

OUTPUT THE FOLLOWING STRING

ASCII /

"LEARN STUDENT REPORT" ( SUMMARY FORM )
<table>
<thead>
<tr>
<th>RPT</th>
<th>MACRO 47 (204)-7 28138 2-AUG-74 PAGE 1-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>962</td>
<td>01361 101 122 116 040 123</td>
</tr>
<tr>
<td>963</td>
<td>01360 124 129 104 105 116</td>
</tr>
<tr>
<td>964</td>
<td>01361 117 129 124 042 048</td>
</tr>
<tr>
<td>965</td>
<td>01361 040 050 060 123 131</td>
</tr>
<tr>
<td>966</td>
<td>01364 106 117 122 119</td>
</tr>
<tr>
<td>967</td>
<td>01365 040 050 015 012 019</td>
</tr>
</tbody>
</table>

**STUDENTS** /

| 968 | 01341 012 015 012 013 014 |
| 969 | 01341 129 184 109 114 124 |
| 970 | 01341 123 372 040 020 000 |

MOVE A, PPN1  
JSP Q.OUTPPN  
ISTRT PROGRAMMER NO. SPECIFIED IN REPORT COMMAND

| 971 | 01341 205 21 0 0 000732 |
| 972 | 01341 265 17 0 0 001271 |
| 973 | 01341 265 17 0 0 001344 |

**LESSON** /

| 974 | 01341 215 212 114 185 123 |
| 975 | 01341 125 117 116 073 041 |
| 976 | 01341 205 205 000 000 000 |

MOVE D, LESSON  
JSP Q.OUTLES  
GET THE LESSON NAME IN AC-D

| 977 | 01341 265 17 0 0 000751 |
| 978 | 01341 265 17 0 0 001361 |
| 979 | 01341 265 17 0 0 000751 |
| 980 | 01341 265 17 0 0 000751 |
| 981 | 01341 265 17 0 0 000751 |
| 982 | 01341 265 17 0 0 000751 |
| 983 | 01341 265 17 0 0 000751 |
| 984 | 01341 265 17 0 0 000751 |
| 985 | 01341 265 17 0 0 000751 |
| 986 | 01341 265 17 0 0 000751 |
| 987 | 01341 265 17 0 0 000751 |

**DATE** /

| 988 | 01341 265 17 0 0 000751 |
| 989 | 01341 265 17 0 0 000751 |
| 990 | 01341 265 17 0 0 000751 |
| 991 | 01341 265 17 0 0 000751 |
| 992 | 01341 265 17 0 0 000751 |
| 993 | 01341 265 17 0 0 000751 |
| 994 | 01341 265 17 0 0 000751 |
| 995 | 01341 265 17 0 0 000751 |
| 996 | 01341 265 17 0 0 000751 |

**TIME** /

| 997 | 01341 265 17 0 0 000751 |
| 998 | 01341 265 17 0 0 000751 |
| 999 | 01341 265 17 0 0 000751 |

**LAST UPDATE** /

| 1000 | 01341 124 124 124 124 124 |
| 1001 | 01341 124 124 124 124 124 |
| 1002 | 01341 124 124 124 124 124 |

JSP Q.OUTTIM  
GET THE CURRENT TIME

**DATE RANGE** /

| 1003 | 01341 015 012 014 012 014 |
| 1004 | 01341 105 040 122 101 116 |
MOVE A, DAY+1  GET THE 1ST DATE SPECIFIED IN THE REPORT COMMAND

JSP Q,ODAT2  OUTPUT THE DATE

JSP Q,OUTSTR  OUTPUT THE FOLLOWING STRING

MOVE A, DAY+1  GET THE 2ND DATE SPECIFIED IN THE REPORT COMMAND

JSP Q,ODAT2  OUTPUT THE 2ND DATE

JSP Q,OUTSTR  OUTPUT THE FOLLOWING STRING

NOW NO CPF EXTOT IDID STUDENT EXECUTE ANY LESSONS?

JSP Q,ODAT2  OUTPUT THE IDID, GO TO SKPOUT

JSP Q,OUTSTR  OUTPUT THE FOLLOWING STRING

ASCII /

TOTAL ELOPSAED TIME/

JSP Q,ODAT2  OUTPUT THE NO.

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /

ASCII /
MOVE A, WGTOT
GET FRACTIONAL PART

MOVE A, PPNL1
PUT PROG. NO. IN AC-A

CAMEL A, PPNL4
HAVE ALL STUDENTS BEEN INCLUDED?

JRPST ALLOUT
YES, GO TO ALLOUT

SETZ EXTOT
SET NO. OF LESSONS EXECUTED TO ZERO

SETZ ANSTOT
CLEAR THE TOTAL ANSWERING TIME

SETZ TOTAL NO. OF RESPONSES

SETZ RTTOT
CLEAR NO. OF CORRECT RESPONSES

SETZ WGTOT
CLEAR NO. OF WRONG RESPONSES

SETZ NTTOT
CLEAR NO. OF NEUTRAL RESPONSES

SETZ TUPTOT
CLEAR NO. OF TIME UPS

SETZ RUNTIM
CLEAR THE TOTAL CPU TIME

SETZ TIMTOT
CLEAR THE TOTAL ELAPSED TIME

USEST 1.1
RETURN TO BEGINNING OF MASTER FILE

JSP Q, INCHR
INPUT THE UPDATE TIME

JSP Q, INCHR
INPUT THE UPDATE DATE

JST FNOBEG
GO TO FNOBEG (SECTION D)

ALLOUT
JSP Q, OUTSTR
GO TO FNOBEG (SECTION D)

NO. OF STUDENTS EXECUTING LESSONS

MOVE A, STDTOT
GET TOTAL NO. OF STUDENTS WHO EXECUTED LESSONS

JSP Q, 100
OUTPUT THE NO.

JSP Q, OUTSTR
OUTPUT THE FOLLOWING STRING

ASC12 /
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
MOVER A: LIM1
MOVER BOTTML: LIM2
PUSH 14: GSORT
MOVER TOP1: TOP
ADS TOP2
MOVER TOP1: LIM1
MOVER B: LIM2
PUSH 14: GSORT
POP 15: B
POP 15: TOP
POP 15: BOTTM1
POP 15: TOP
POP 15: BOTTM
POP 15: TOP
POP 15: B
POP 14,
END1: SETEAG:
PARAM-1
GOBACK27
MILLS A: PARAM(G) GET FRAME NO. FROM RETRIEVAL LIST
JMP A: ALLFIN IF END OF LIST, GO TO ALLFIN
MILLS PARAM(1-G) CLEAR RIGHT HALF OF PRECEDING ITEM
HREC C: PARAM(G) SET CHARACTER COUNT IN AC-C
CAMG C: PARAM-1(G) IS NEW FRAME NO. SAME AS LAST FRAME NO.?
JST NOP LINES YES, GO TO NOP LINES
JSP G: OUTSTA ELSE, OUTPUT THE FOLLOWING STRING
ASCIS /

NOLINS1: TRUE FLAG5:2
IS LISTING TO LPT?

JST *+4 INC. JUMP
MOVER A:11 PUT TAB IN AC-A
JSP G: OUTCHR OUTPUT TAB
JSP G: OUTCHR OUTPUT TAB
IDIV C:1280 DIVIDE CHARACTER COUNT BY NO. OF CHARACTERS PER BLOCK
ADS C ADD ONE TO GET DISK BLOCK CONTAINING THE LINE
USETI 1(C)
INPUT 1
MOVER A:BUFF-2 GET NO. OF CHARACTERS IN THE BLOCK
SUB A:JOP COMPUTE NO. OF CHARACTER AFTER BEGINNING OF THE LINE
MOVER A:BUFF-2 SET CHARACTER COUNT EQUAL TO THIS NO.
IDIV D:5 COMPUTE WORD POINT FOR THE LINE
ADDM D:8BUFF-1 ADD IT TO BUFFER BYTE POINTER
JUMPE 1:13 IF POINTER IS AT BEGINNING OF LINE, JUMP
JSP G: OUTCHR INCENT BYE POINTER
JSP G: INCHR INPUT FRAME NO. DIGIT
JSP G: OUTCHR OUTPUT FRAME NO. DIGIT
MOVER A:13 GO INPUT NEXT DIGIT, IF ANY
MOVER A:11
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
PROCESSED LESSON NOT FOUND

JRS T 0PC1

RETURN TO LRPRT, MAC

DELLALE IRE FLAGS, Z CLEAR OTHER FILE FLAG

INSTR FLAGS, CLEAR FLAG AND JUMP

DELTRAN IRE FLAGS, 3 CLEAR MASTER FILE AND OTHER NAME FLAGS

JRS TRNCH IIF PROG. NO. WAS SPECIFIED, GO TYPE ERROR MESSAGE

JRS DTRN IIF DTH DATE WAS SPECIFIED, GO TYPE ERROR MESSAGE

SKIPN FRAME 1

JRS TRNCH IIF FRAME NO. WAS SPECIFIED, GO TYPE ERROR MESSAGE

TRN CH EXISTING MASTER FILE FLAG SET?

MOVE A, IINAH IYES, GET MASTER FILENAME

TRN CH EXISTING MASTER FILE CLEAR?

MOVE A, IINAH IYES, PUT TRANSACTION FILENAME IN AC-A

MOVCH A, IINAH IPUT FILENAME IN INAH

OVRNCH MOV CH A, IINAH IPUT REC IN INAH

SET INAH 3

LOOKUP 1, INAH ILOOKUP RECORD FILE

JRS NOTRN IIF NOT FOUND, GO TYPE ERROR MESSAGE

SET INAH 3

SET PROTECTION TO 168

MOVE A, IINAH 2

RENAME 1, IINAH ICHANGE PROTECTION SO FILE CAN BE DELETED

WALT

SET INAH 3

LOOKUP 1, INAH ILOOKUP RECORD FILE AGAIN

WALT

SET INAH

RENAME 1, IINAH IDELETE RECORD FILE

WALT

WALT

TRN CH EXISTING OTHER FILE FLAG SET?

JRS ORROE IYES, GO TO ORROE

TRN CH EXISTING MASTER FILE FLAG SET?

JRS MASDE IYES, GO TO MASDE

TTCALL 3, IASC12 / TRANSACTION FILE DELETED

JRS 0PC1

RETURN TO LRPRT, MAC

ORROE TDCALL 3, IASC12 / RECORD FILE DELETED

MASDE TDCALL 3, IASC12 / MASTER FILE DELETED

JRS 0PC1

RETURN TO LRPRT, MAC
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES31</td>
<td>TDCALL 4, AINPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>CAIE 1.12</td>
<td>IEND OF LINE?</td>
</tr>
<tr>
<td>JRS1.2</td>
<td>INO, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>OPEN 3,0,PP3</td>
<td></td>
</tr>
<tr>
<td>HALT</td>
<td></td>
</tr>
<tr>
<td>MOVE 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOVE 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,RECO1</td>
<td></td>
</tr>
<tr>
<td>MOVE 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,RECO1</td>
<td></td>
</tr>
<tr>
<td>SET3M 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>LOOKUP 3,0,NAH</td>
<td></td>
</tr>
<tr>
<td>ILOOKUP TRANSACTION FILE</td>
<td></td>
</tr>
<tr>
<td>JRS1.6</td>
<td>IJUMP IF NOT FOUND</td>
</tr>
<tr>
<td>SET3M 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>SET3M 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>SET3M 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>RENAME 3,0,NAH</td>
<td></td>
</tr>
<tr>
<td>CHANGE PROTECTION SO FILE CAN BE SUPERSEDED</td>
<td></td>
</tr>
<tr>
<td>JRS1 BUSY2</td>
<td>IGO TO BUSY2 IF FILE IS BEING MODIFIED</td>
</tr>
<tr>
<td>HLLES 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>MOV1 1,0,NAH1</td>
<td></td>
</tr>
<tr>
<td>SET3M 0,NAH1</td>
<td></td>
</tr>
<tr>
<td>ENTER 3,0,NAH</td>
<td></td>
</tr>
<tr>
<td>ENTER NEW TRANSACTION FILE</td>
<td></td>
</tr>
<tr>
<td>JRS1 BUSY2</td>
<td>IGO TO BUSY2 IF FILE IS BEING MODIFIED</td>
</tr>
<tr>
<td>TRANSACTION FILE CLEARED</td>
<td></td>
</tr>
<tr>
<td>RELEASE 3,</td>
<td></td>
</tr>
<tr>
<td>JRS1 PC10</td>
<td>IRETURN TO LRPRRT,MAC</td>
</tr>
<tr>
<td>BUSY21</td>
<td>HARE B,0,NAH+1 IGET ERROR CODE</td>
</tr>
<tr>
<td>JRS1 BUSY2</td>
<td>IIS FILE BEING MODIFIED</td>
</tr>
<tr>
<td>JRS1 NOROM</td>
<td>INO, GO TO NOROM</td>
</tr>
<tr>
<td>MOVE 0,1</td>
<td></td>
</tr>
<tr>
<td>SLEEP 0,</td>
<td>ISLEEP A SECOND</td>
</tr>
<tr>
<td>JRS1 AGNN</td>
<td>IGO TRY AGAIN</td>
</tr>
<tr>
<td>NOROM</td>
<td>TTCALL 3, ASCIIZ/</td>
</tr>
<tr>
<td>NO ROOM ON DISK</td>
<td></td>
</tr>
<tr>
<td>CLOSE 2,40</td>
<td>IKEEP ORIGINAL FILE</td>
</tr>
<tr>
<td>RELEASE 3,</td>
<td></td>
</tr>
<tr>
<td>RELEASE 3,</td>
<td></td>
</tr>
<tr>
<td>RELEASE 4,</td>
<td></td>
</tr>
<tr>
<td>SETSTS 1,0</td>
<td>IRESTORE ASCII MODE</td>
</tr>
<tr>
<td>SETSTS 2,0</td>
<td>IRESTORE ASCII MODE</td>
</tr>
<tr>
<td>MOVE 0,0</td>
<td>IRESTORE ASCII BYTE POINTER</td>
</tr>
<tr>
<td>MOVE 0,0</td>
<td>IRESTORE ASCII BYTE POINTER</td>
</tr>
<tr>
<td>JRS1 PC10</td>
<td>IRETURN TO LRPRRT,MAC</td>
</tr>
</tbody>
</table>

**SECTION 1: EXECUTE UPDATE COMMANDS:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDT1</td>
<td>MOVE Q,PC1 ISAVE RETURN ADDRESS</td>
</tr>
<tr>
<td>TTCALL 4, A</td>
<td>IINPUT CHARACTER FROM TERMINAL</td>
</tr>
<tr>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>CHKFR</td>
<td>1726</td>
</tr>
<tr>
<td>CLB</td>
<td>1726</td>
</tr>
<tr>
<td>DLRTN</td>
<td>1726</td>
</tr>
<tr>
<td>CMERR</td>
<td>1726</td>
</tr>
<tr>
<td>D</td>
<td>1726</td>
</tr>
<tr>
<td>DALL</td>
<td>1672</td>
</tr>
<tr>
<td>DATA</td>
<td>1672</td>
</tr>
<tr>
<td>DATAQK</td>
<td>1672</td>
</tr>
<tr>
<td>DATEQK</td>
<td>1672</td>
</tr>
<tr>
<td>DATER</td>
<td>1672</td>
</tr>
<tr>
<td>DATERQ</td>
<td>1672</td>
</tr>
<tr>
<td>DATO</td>
<td>1672</td>
</tr>
<tr>
<td>DATQ</td>
<td>1672</td>
</tr>
<tr>
<td>DEL</td>
<td>1672</td>
</tr>
<tr>
<td>DEL3</td>
<td>1672</td>
</tr>
<tr>
<td>DELALL</td>
<td>1672</td>
</tr>
<tr>
<td>DELC</td>
<td>1672</td>
</tr>
<tr>
<td>DELHAS</td>
<td>1672</td>
</tr>
<tr>
<td>DELPR</td>
<td>1672</td>
</tr>
<tr>
<td>DELPRK</td>
<td>1672</td>
</tr>
<tr>
<td>DELT</td>
<td>1672</td>
</tr>
<tr>
<td>DEVR</td>
<td>1672</td>
</tr>
<tr>
<td>DEVRE</td>
<td>1672</td>
</tr>
<tr>
<td>DEVIN</td>
<td>1672</td>
</tr>
<tr>
<td>DEVOX</td>
<td>1672</td>
</tr>
<tr>
<td>DFMS</td>
<td>1672</td>
</tr>
<tr>
<td>DNL</td>
<td>1672</td>
</tr>
<tr>
<td>DOWN</td>
<td>1672</td>
</tr>
<tr>
<td>DSWT</td>
<td>1672</td>
</tr>
<tr>
<td>END2</td>
<td>1672</td>
</tr>
<tr>
<td>ENDFIL</td>
<td>1672</td>
</tr>
<tr>
<td>ENDFL</td>
<td>1672</td>
</tr>
<tr>
<td>ENDSL</td>
<td>1672</td>
</tr>
<tr>
<td>EOFDEL</td>
<td>1672</td>
</tr>
<tr>
<td>EOFDE</td>
<td>1672</td>
</tr>
<tr>
<td>EXTOT</td>
<td>1672</td>
</tr>
<tr>
<td>F</td>
<td>1672</td>
</tr>
<tr>
<td>FLGS</td>
<td>1672</td>
</tr>
<tr>
<td>FND</td>
<td>1672</td>
</tr>
<tr>
<td>FNDSEG</td>
<td>1672</td>
</tr>
<tr>
<td>FMN</td>
<td>1672</td>
</tr>
<tr>
<td>FRM</td>
<td>1672</td>
</tr>
<tr>
<td>FRAME</td>
<td>1672</td>
</tr>
<tr>
<td>FNAME</td>
<td>459</td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
</tr>
<tr>
<td>G</td>
<td>253#</td>
</tr>
<tr>
<td></td>
<td>1293</td>
</tr>
<tr>
<td></td>
<td>1706</td>
</tr>
<tr>
<td>GETDA</td>
<td>715#</td>
</tr>
<tr>
<td>GETDAT</td>
<td>292#</td>
</tr>
<tr>
<td>GETEND</td>
<td>1516#</td>
</tr>
<tr>
<td>GETFRM</td>
<td>331</td>
</tr>
<tr>
<td>GETINF</td>
<td>162#</td>
</tr>
<tr>
<td>GETN</td>
<td>326</td>
</tr>
<tr>
<td>GETN2</td>
<td>356#</td>
</tr>
<tr>
<td>GETNUM</td>
<td>293</td>
</tr>
<tr>
<td>GETSWT</td>
<td>326</td>
</tr>
<tr>
<td>G2N</td>
<td>393#</td>
</tr>
<tr>
<td>G2NM</td>
<td>296#</td>
</tr>
<tr>
<td>G2CK2</td>
<td>1297#</td>
</tr>
<tr>
<td>G2TALL</td>
<td>323</td>
</tr>
<tr>
<td>G2TFRM</td>
<td>457</td>
</tr>
<tr>
<td>G2G1</td>
<td>1660</td>
</tr>
<tr>
<td>G2G2</td>
<td>687</td>
</tr>
<tr>
<td>G2G3</td>
<td>242#</td>
</tr>
<tr>
<td>G2G4</td>
<td>241#</td>
</tr>
<tr>
<td>G2G5</td>
<td>241#</td>
</tr>
<tr>
<td>G2G6</td>
<td>1138</td>
</tr>
<tr>
<td>G2G7</td>
<td>1143</td>
</tr>
<tr>
<td>G2G8</td>
<td>1153</td>
</tr>
<tr>
<td>II</td>
<td>240#</td>
</tr>
<tr>
<td>II2</td>
<td>258#</td>
</tr>
<tr>
<td>II2SWT</td>
<td>401</td>
</tr>
<tr>
<td>INAM</td>
<td>241#</td>
</tr>
<tr>
<td></td>
<td>1096</td>
</tr>
<tr>
<td></td>
<td>1428</td>
</tr>
<tr>
<td></td>
<td>1473</td>
</tr>
<tr>
<td></td>
<td>1937</td>
</tr>
<tr>
<td>INAM2</td>
<td>241#</td>
</tr>
<tr>
<td></td>
<td>1125</td>
</tr>
<tr>
<td>INCHR</td>
<td>242#</td>
</tr>
<tr>
<td></td>
<td>1165</td>
</tr>
<tr>
<td>INPUT</td>
<td>241#</td>
</tr>
<tr>
<td>INWRT</td>
<td>241#</td>
</tr>
<tr>
<td>IOO</td>
<td>239#</td>
</tr>
<tr>
<td></td>
<td>908</td>
</tr>
<tr>
<td>KEEPFM</td>
<td>1711</td>
</tr>
<tr>
<td>LESSND</td>
<td>641</td>
</tr>
<tr>
<td>LESSON</td>
<td>243#</td>
</tr>
<tr>
<td></td>
<td>1446</td>
</tr>
<tr>
<td>LIM1</td>
<td>258#</td>
</tr>
<tr>
<td>LIM2</td>
<td>258#</td>
</tr>
<tr>
<td>LRT</td>
<td>258#</td>
</tr>
<tr>
<td>LTRPT</td>
<td>1531</td>
</tr>
<tr>
<td>MONTH</td>
<td>240#</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>NA1</td>
<td>258#</td>
</tr>
<tr>
<td>NA2</td>
<td>259#</td>
</tr>
<tr>
<td>NAERR</td>
<td>373</td>
</tr>
<tr>
<td>NAMIN</td>
<td>396</td>
</tr>
<tr>
<td>NLH</td>
<td>1757</td>
</tr>
<tr>
<td>MNIN</td>
<td>1425</td>
</tr>
<tr>
<td>NODELT</td>
<td>1385#</td>
</tr>
<tr>
<td>NOFIL2</td>
<td>1902</td>
</tr>
<tr>
<td>NOLINS</td>
<td>1302</td>
</tr>
<tr>
<td>NOLUCK</td>
<td>1098</td>
</tr>
<tr>
<td>NODST</td>
<td>1347</td>
</tr>
<tr>
<td>NODTR</td>
<td>1545#</td>
</tr>
<tr>
<td>NORORH</td>
<td>620</td>
</tr>
<tr>
<td>NOTRAN</td>
<td>616</td>
</tr>
<tr>
<td>NOTRN</td>
<td>1515</td>
</tr>
<tr>
<td>NTTOT</td>
<td>242#</td>
</tr>
<tr>
<td>NULL</td>
<td>1113</td>
</tr>
<tr>
<td>NUM2</td>
<td>364</td>
</tr>
<tr>
<td>NUMERR</td>
<td>393#</td>
</tr>
<tr>
<td>NUMIN</td>
<td>358</td>
</tr>
<tr>
<td>OBUF</td>
<td>243#</td>
</tr>
<tr>
<td>OBUF2</td>
<td>241#</td>
</tr>
<tr>
<td>ONAH</td>
<td>243#</td>
</tr>
<tr>
<td>OPA3</td>
<td>259#</td>
</tr>
<tr>
<td>OPA4</td>
<td>1535</td>
</tr>
<tr>
<td>OPP3</td>
<td>1553#</td>
</tr>
<tr>
<td>OTRDAT</td>
<td>239#</td>
</tr>
<tr>
<td>OTRDEL</td>
<td>1539</td>
</tr>
<tr>
<td>OTRNAM</td>
<td>1407</td>
</tr>
<tr>
<td>OTTIM2</td>
<td>239#</td>
</tr>
<tr>
<td>OUTCHR</td>
<td>240#</td>
</tr>
<tr>
<td>OUTFRE</td>
<td>1235</td>
</tr>
<tr>
<td>OUTG</td>
<td>1344</td>
</tr>
<tr>
<td>OUTDAT</td>
<td>239#</td>
</tr>
<tr>
<td>OUTLES</td>
<td>239#</td>
</tr>
<tr>
<td>OUTPPN</td>
<td>239#</td>
</tr>
<tr>
<td>OUTSTR</td>
<td>239#</td>
</tr>
<tr>
<td>OUTTOT</td>
<td>239#</td>
</tr>
<tr>
<td>OUTUS</td>
<td>1044</td>
</tr>
<tr>
<td>OUTVIN</td>
<td>239#</td>
</tr>
<tr>
<td>OUTUM</td>
<td>239#</td>
</tr>
<tr>
<td>OUTWHRD</td>
<td>241#</td>
</tr>
</tbody>
</table>
TITLE LPTRPT

COMMENT

WRITTEN BY GARRET A. VANDER LUST  COMPLETED 9-FEB-74

PART OF LRPRT.SH, 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= A, LOCATED IN AUX3.MAC (SECTION B)

IOD - OUTPUTS THE INTEGER IN AC=A TO THE DISK, LOCATED IN AUX3.MAC (SECTION D)

ODTDEZ - OUTPUTS THE DATE IN AC=A TO THE DISK, LOCATED IN AUX3.MAC (SECTION J)

OTTIM2 - OUTPUTS THE TIME IN AC=A 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)
OUTPPN = 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 NOS. SPECIFIED IN REPORT COMMAND (2 WORDS)
IBUF = DISK BUFFER HEADER (13 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 NO. SPECIFIED IN REPORT COMMAND (2 WORDS)

RESTOT = TOTAL NO. OF RESPONSES IS ACCUMULATED HERE

RTTOT = TOTAL NO. OF CORRECT RESPONSES IS ACCUMULATED HERE

RUNTIM = TOTAL CPU TIME IS ACCUMULATED HERE

TIMITOT = 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 29 STUDENT REPORT

ENTRY LPRPT

EXTERN TIMTOT, TUITOT, WGTOT, PARAM, FRAME

EXTERN ANSTOT, IDO, NTTOT, OUTTH, RESTOT, RTJUST, RTTOT, RUNTIM

EXTERN FCL, ABO, DAY1, INCHR, LESSON, OTTIM, OUTTIM, ODTD1, ODTD2, OUTDAT

EXTERN OUTCHR, OUTLES, OUTPPN, OUTSTR, PPNI, EXTOT, RPRINT, DATA, IBUF

FLAGS=0

000000

000001

000002

000003

000004

000005

000006

000007

000008

000009

00000A

00000B

00000C

00000D

00000E

00000F

000010

000011

000012

000013

000014

000015

000016

000017

000018

RECLST=PARAM+1

FMTDAT=RECLST+1713

SECTION A. WRITE THE REPORT HEADING.

LPRPTI MOVE B, ABO GET THE REPORT COMMAND SWITCH

LPRPTI MOVIE A, 777 GET THIS IS A STUDENT REPORT?

LPRPTI 214 0 0 003467 YES, GO TO STOPRT (SECTION C)

LPRPTI 620 0 0 000100 TRE FLAGS=100 ELSE, CLEAR THE STUDENT REPORT FLA
SECTION C. GET THE FRAME DATA FROM MASTER,REC.

...
SECTION D. WRITE THE SUMMARY DATA ON THE DISK.

ENDFILL JSP Q;OUTSTR 1OUTPUT THE FOLLOWING STRING
ASCIE /

NO. TIMES LESSON WAS EXECUTED /

MOVE A,EXTOT 1GET THE NO. OF TIMES THE LESSON WAS EXECUTED

JSP Q,100 1OUTPUT THE NO.

SKIP E Extot 1WAS LESSON EXECUTED AT LEAST ONCE?

JSP Q,LesfnD 1YES, GO TO LesfnD

JSP Q;OUTSTR 1ELSE, OUTPUT THE FOLLOWING STRING
ASCIE /

TRNE FLAGS,100,1S THIS A STUDENT REPORT?

JST StDR2 1YES, GO TO StDR2 (SECTION G)

JST reset 1ELSE, GO TO RESET (SECTION F)

JSP Q;OUTSTR 1OUTPUT THE FOLLOWING STRING
ASCIE /

AVE. ELAPSED TIME1 /

MOVE A,TIMTOT 1GET THE TOTAL ELAPSED TIME

DIV A,Extot 1divide by no. of times lesson was executed

SETZ C 1SET AC-C TO ZERO FOR THE OUTM SUBROUTINE

JSP Q;OUTSTR 1OUTPUT THE ELAPSED TIME

JSP Q;OUTSTR 1OUTPUT THE FOLLOWING STRING
ASCIE / MINUTES

AVE. TOTAL ANSWERING TIME1 /
SECTION C. WRITE THE FRAME DATA ON THE DISK.

THE ROUTINE BELOW Sorts THE RECORD LIST IN ORDER OF INCREASING FRAME NO.

LABEL FRAME NO.
<table>
<thead>
<tr>
<th>LPTRT MACRO 47(204)-7 00141</th>
<th>2-AUG-74 PAGE 1-14</th>
</tr>
</thead>
<tbody>
<tr>
<td>591 001021 400 0 0 0 001000</td>
<td>SETZ F.</td>
</tr>
<tr>
<td>592 001022 555 0 0 0 001000</td>
<td>GOB21</td>
</tr>
<tr>
<td>593 001023 322 0 0 0 001000</td>
<td>HRTZ C,RECLST(F)</td>
</tr>
<tr>
<td>594 001024 365 0 0 0 001050</td>
<td>GERT FRM NO. FROM RECORD LIST</td>
</tr>
<tr>
<td>595 001025 254 0 0 0 001043</td>
<td>JUMP C,ENDTFOH END OF LIST, GO TO ENDTFOH</td>
</tr>
<tr>
<td>596 001026 275 0 0 0 001070</td>
<td>CAIGE C,01050</td>
</tr>
<tr>
<td>597 001027 602 0 0 0 001032</td>
<td>IS THIS A REVIEW FRAME?</td>
</tr>
<tr>
<td>598 001030 254 0 0 0 001031</td>
<td>JRST NOTREV</td>
</tr>
<tr>
<td>599 001031 605 0 0 0 001032</td>
<td>NO, GO TO NOTREV</td>
</tr>
<tr>
<td>600 001032 265 17 0 0 001071</td>
<td>SUBC C,010301</td>
</tr>
<tr>
<td>601 001033</td>
<td>ELSE, SUBTRACT 1000 FROM FRAME NO.</td>
</tr>
<tr>
<td>602 001034</td>
<td>TRN FLAGS,2001</td>
</tr>
<tr>
<td>603 001035</td>
<td>IS REVIEW FRAME FLAG SET?</td>
</tr>
<tr>
<td>604 001036</td>
<td>YES, GO TO NOTREV</td>
</tr>
<tr>
<td>605 001037</td>
<td>TRF FLAGS,2001</td>
</tr>
<tr>
<td>606 001038</td>
<td>ELSE, SET REVIEW FRAME FLAG</td>
</tr>
<tr>
<td>607 001039</td>
<td>JSP G,OUTSTR</td>
</tr>
<tr>
<td>608 001040</td>
<td>OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>609 001041</td>
<td>ASCIE</td>
</tr>
<tr>
<td>610 001042</td>
<td>REVIEW FRAME DATA</td>
</tr>
<tr>
<td>611 001043</td>
<td>NOTREV: JSP G,OUTSTR</td>
</tr>
<tr>
<td>612 001044</td>
<td>OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>613 001045</td>
<td>ASCIE</td>
</tr>
<tr>
<td>614 001046</td>
<td>FRAME.</td>
</tr>
<tr>
<td>615 001047</td>
<td>MOVE A,C</td>
</tr>
<tr>
<td>616 001048</td>
<td>INPUT FRAME NO. IN AC-A</td>
</tr>
<tr>
<td>617 001049</td>
<td>JSP G,100</td>
</tr>
<tr>
<td>618 001050</td>
<td>OUTPUT THE FRAME NO.</td>
</tr>
<tr>
<td>619 001051</td>
<td>JSP G,OUTSTR</td>
</tr>
<tr>
<td>620 001052</td>
<td>OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>621 001053</td>
<td>ASCIE</td>
</tr>
<tr>
<td>622 001054</td>
<td>TYPE -</td>
</tr>
<tr>
<td>623 001055</td>
<td>LDB A,POINT 7,RECLST(F),6</td>
</tr>
<tr>
<td>624 001056</td>
<td>IGET THE FRAME TYPE</td>
</tr>
<tr>
<td>625 001057</td>
<td>JSP G,OUTCHR</td>
</tr>
<tr>
<td>626 001058</td>
<td>OUTPUT FRAME TYPE</td>
</tr>
<tr>
<td>627 001059</td>
<td>JSP G,OUTSTR</td>
</tr>
<tr>
<td>628 001060</td>
<td>OUTPUT THE FOLLOWING STRING</td>
</tr>
<tr>
<td>629 001061</td>
<td>ASCIE</td>
</tr>
<tr>
<td>630 001062</td>
<td>NO. OF TIMES FRAME WAS EXECUTED.</td>
</tr>
<tr>
<td>631 001063</td>
<td>LDB A,POINT 11,RECLST(F),17</td>
</tr>
<tr>
<td>632 001064</td>
<td>IGET THE DATA INDEX FROM THE RECORD LIST</td>
</tr>
<tr>
<td>633 001065</td>
<td>IMULI G,16</td>
</tr>
<tr>
<td>634 001066</td>
<td>MULTIPLY BY NO. OF DATA WORDS FOR EACH FRAME</td>
</tr>
<tr>
<td>635 001067</td>
<td>HRTZ A,FRMDAT(S)</td>
</tr>
<tr>
<td>636 001068</td>
<td>IGET NO. OF TIMES FRAME WAS EXECUTED</td>
</tr>
<tr>
<td>637 001069</td>
<td>MOVEM A,PARAM</td>
</tr>
<tr>
<td>638 001070</td>
<td>SAVE IT</td>
</tr>
<tr>
<td>639 001071</td>
<td>JSP G,100</td>
</tr>
<tr>
<td>640 001072</td>
<td>OUTPUT NO. OF TIMES FRAME WAS EXECUTED</td>
</tr>
</tbody>
</table>

* Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>LPLTRPT</th>
<th>MACRO 47(204)-7 00141 2-AUG-74 PAGE 1-17</th>
</tr>
</thead>
<tbody>
<tr>
<td>001251</td>
<td>00 1251 1.24 161 114 040 116</td>
</tr>
<tr>
<td>001252</td>
<td>117 056 043 117 106</td>
</tr>
<tr>
<td>001253</td>
<td>040 124 111 115 109</td>
</tr>
<tr>
<td>001254</td>
<td>140 125 120 132 072</td>
</tr>
<tr>
<td>001255</td>
<td>040 040 122 040 105</td>
</tr>
<tr>
<td>001256</td>
<td>140 122 117 122 040</td>
</tr>
<tr>
<td>001257</td>
<td>122 101 124 105 072</td>
</tr>
<tr>
<td>001258</td>
<td>043 029 000 029 003</td>
</tr>
<tr>
<td>001259</td>
<td>200 200 000 000 033</td>
</tr>
<tr>
<td>001260</td>
<td>MOVE B,DATA=3</td>
</tr>
<tr>
<td>001261</td>
<td>ADD B,DATA=2</td>
</tr>
<tr>
<td>001262</td>
<td>MOVE A,DATA=3</td>
</tr>
<tr>
<td>001263</td>
<td>IMUL I,DOS000</td>
</tr>
<tr>
<td>001264</td>
<td>IDIV A,DOS10</td>
</tr>
<tr>
<td>001265</td>
<td>IDIV A,DOS10</td>
</tr>
<tr>
<td>001266</td>
<td>CALL B,5</td>
</tr>
<tr>
<td>001267</td>
<td>AOS A</td>
</tr>
<tr>
<td>001268</td>
<td>IDIV A,DOS5</td>
</tr>
<tr>
<td>001269</td>
<td>MOVE B,DATA=3</td>
</tr>
<tr>
<td>001270</td>
<td>JSP Q,100</td>
</tr>
<tr>
<td>001271</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>001272</td>
<td>ASC12 */</td>
</tr>
<tr>
<td>001273</td>
<td>MOVE A,DATA=3</td>
</tr>
<tr>
<td>001274</td>
<td>JSP Q,100</td>
</tr>
<tr>
<td>001275</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>001276</td>
<td>ASC12 */</td>
</tr>
<tr>
<td>001277</td>
<td>RESPONSE DISTRIBUTION</td>
</tr>
<tr>
<td>001278</td>
<td>MOVE A</td>
</tr>
<tr>
<td>001279</td>
<td>AOS G</td>
</tr>
<tr>
<td>001280</td>
<td>G0BK41</td>
</tr>
<tr>
<td>001281</td>
<td>JR2 A,FRMSTD(G)</td>
</tr>
<tr>
<td>001282</td>
<td>JUMPE A,GOBK3</td>
</tr>
<tr>
<td>001283</td>
<td>JSP Q,OUTCHR</td>
</tr>
<tr>
<td>001284</td>
<td>JSP Q,OUTSTR</td>
</tr>
<tr>
<td>001285</td>
<td>ASC12 */</td>
</tr>
<tr>
<td>001286</td>
<td>HR2 A,FRMSTD(G)</td>
</tr>
<tr>
<td>001287</td>
<td>JSP Q,DATA=5</td>
</tr>
<tr>
<td>001288</td>
<td>JSP Q,100</td>
</tr>
<tr>
<td>001289</td>
<td>JSP Q,OUTTAG</td>
</tr>
<tr>
<td>001290</td>
<td>MOVE A,DATA=5</td>
</tr>
<tr>
<td>001291</td>
<td>JSP Q,100</td>
</tr>
<tr>
<td>001292</td>
<td>MOVE A,DATA=5</td>
</tr>
<tr>
<td>001293</td>
<td>CAIGE 9,01000</td>
</tr>
</tbody>
</table>
SECTION F: PLACE THE REPORT FILE IN PRINT QUEUE.

753 001354 201 01 00 00 01777  MOVE A, #777
754 001355 047 01 00 00 00001  IREDUCE LOW SEGMENT TO 1K
755 001356 254 04 00 00 00000  MALT
756 001357 070 03 00 00 00000  CLOSE 1:
757 001358 070 02 00 00 00000  CLOSE 2:
758 001359 060 01 00 00 00000  SETSTS L,0
759 001360 200 01 00 00 01432 MOVETOA, #00000000
760 001361 202 01 00 00 00001 MOVETOA, #00000001
761 001362 254 17 00 00 00000 JSPG.OUTSTR
762 001363 015 02 00 00 00000 ASCII /
763 001364 019 02 00 00 00000 ASCII /
764 001365 025 03 00 00 01433 JSPG.OUTSTR
765 001366 254 00 00 00 00230 JRT MAC
766 001367 620 00 00 00 00210 STORPT TITE FLAGS,100 CLEAR STUDENT REPORT FLAG
767 001368 200 02 00 00 00002 MOVETOB, #00000002
768 001369 222 02 00 00 00000 MOVETOB, #00000006
769 001370 254 00 00 00 01374 JRT +2
770 001371 254 00 00 00 00137 TCALL 3,ASCII /

REPORT FILE IN PRINT QUEUE

773 001372 350 00 00 00 00245 STOR TO APS PPN1
774 001373 200 00 00 00 01373 IINCREDMENT PROGRAMMER NO.
775 001374 222 00 00 00 00001 MOVETOB, #00000001
776 001375 313 00 00 00 00002 CAMLEB, #00000002
777 001376 254 00 00 00 01354 JRT RET
778 001377 074 01 00 00 00001 USETJ, L,1
779 001378 201 01 00 00 00014 MOVETOA, #14
780 001379 602 00 00 00 00100 TAKEN100 IC STUDENT REPORT FLAG SET?
NO ERRORS DETECTED

PROGRAM BREAK IS 001442

3K CORE USED
TITLE RPRINT

PREPARED BY GARRET A. VANDER LUGT  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 RPRT.MAC (SECTIONS D AND F) AND LRPRPT.MAC (SECTION F);
LOCATED IN SECTION A

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN RPLLOW.MAC)

EXTOT (=THSJOB) = USER'S JOB NUMBER

RUNTIM (=CEDIR) = NAME OF QUEUING UFD

INAM2 (=OPNBKLK) = ARGUMENTS FOR OPEN UDU (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)

93 8888881

HISEQ
ENTRY RPRINT

EXTERN RUNIT, RUNTIM, INAM2, LESSON, CMOLST, DATA, ONAH, PARAH

OPLIST = CMOLST
OQLSPEC = DATA

A = 1
B = 2
C = 3
D = 4
E = 5
F = 6
G = 7
H = 8
I = 9
J = 10
K = 11
L = 12
M = 13
N = 14
O = 15
P = 16
Q = 17
R = 18
S = 19
T = 20
U = 21
V = 22
W = 23
X = 24
Y = 25
Z = 26

SECTION A. SET UP QUEUING PARAMETERS IN CORE.

OPNI 16
SIXBIT / DSK/

RPRINT MOVE A, [34, 111]

GETTAB A, GET MONITOR VERSION NO.

HALT

CAIGE A, 50300, 1IS MONITOR VERSION 5, 03 OR LATER?

HALT

IND, HALT

OPEN 0, OPN

HALT

MOVEI A, 40

SETEM PARAH-1(A), CLEAR QUEUING PARAMETERS

SOJG A, -=1

MOVEI A, 16

MOVE A, QLSPEC, MOVE 14 TO QLSPEC FOR EXTENDED LOOKUP

MOVE A, QLSPEC+1, MOVE LOOKUP IN USER'S OWN UDF

MOVE A, ONAH, GET FILENAME OF LIST FILE

MOVE A, QLSPEC+2, MOVE IT TO LookUp EFFECTIVE ADDRESS

MOVE A, PARAH+5, MOVE IT TO PARAMETER LIST ALSO

MOVE A, PARAH+33

MOVEI A, LIST

MOVE A, QLSPEC+3, MOVE EXT. TO LOOKUP EFFECTIVE ADDRESS

MOVE A, PARAH+34

OPEN 0, OPN, OPEN CHANNEL 0, DUMP MODE

HALT

LOOKUP 0, QLSPEC 100 LOOKUP ON LIST FILE
SETH A;
MOVEM A,PARAM+15; MOVE IT TO PARAMETER LIST
HRLT A,(B)

GETTAB A; IGET 1ST HALF OF USER'S NAME

MOVEL A,PARAM+16; PUT IT IN PARAMETER LIST
HRLT A,(B)

GETTAB A; IGET 2ND HALF OF USER'S NAME

SETH A;

MOVEM A,PARAM+17; PUT IT IN PARAMETER LIST

ISSECTION B. WRITE PARAMETER FILE ON THE DISK.

MOVE A,(15,16); "GET FILE STRUCTURE CONTAINING SPOOLING UFD

GETTAB A;

MOVEL A,DISK; MOVE B,(4,16)

GETTAB A;

MOVEL B,(3,31)

MOVEL C,(480000,171)

GETTAB B; IGET PPN FOR SPOOLING PROGRAMS

MOVE D,(1,163)

MOVE D,(1,153)

GETTAB B; IGET PPN FOR DEVICE SYS

MOVE D,(1,153)

MOVE B,QUEDIR

MOVEL D,OPNLK

SETH D,OPNLK+2

MOVEL A,OPNLK+1

OPEN B,OPNLK

HALT

MOVEL B,LP'T;

WHERE B;

GETSTATION NO. OF LINE PRINTER

SETH B;

HRLT B,LP'T;

MOVEM B,PARAM+3

IFIND SUITABLE NAME FOR PARAMETER FILE

MSTIME A;

IDIVI A,*0100

CRENHI MOVEL B,LP'T

MOVE C,(POINT6,B,11)

ADD A,TH508

MOVE D,A

MOVEL D,IDIVI D,*010

ADDI E,*01

IDPB E,C

TLNE 0,(7709) IDONE YET?

JST CRENHI END

MOVEL D,C,DQUE!

MOVEL B,FILNM SAVE FILENAME

MOVEL B,QUEDIR
I

\[ \text{LOOKUP O,B \ Y \ IS \ NAME \ IN \ USE?} \]
\[ \text{JST CRENH \ YES, \ GO \ TO \ CRENH} \]
\[ \text{JUMP C,CRENH} \]
\[ \text{MOVE B,FILNAM} \]
\[ \text{MOVS I,C,QUE} \]
\[ \text{MOVE D,177000} \]
\[ \text{MOVE E,QUEDIR} \]
\[ \text{CLOSE O} \]
\[ \text{ENTER O,B \ ENTER \ PARAMETER \ FILE} \]
\[ \text{MALT} \]
\[ \text{MOVE A,[CROWD 37,PARAMA]} \]
\[ \text{MOVE M,OLIST \ SET \ OLIST} \]
\[ \text{OUTPUT O,OLIST \ OUTPUT \ QUEUING \ PARAMETERS} \]
\[ \text{CLOSE O} \]
\[ \text{RELEASE O} \]
\[ \text{SETZ B} \]

\[ \text{SECTION C \ WAKE \ UP \ THE \ SPOOLING \ PROGRAM} \]

\[ \text{CREDNO \ AOS B} \]
\[ \text{HRLZ C,B} \]
\[ \text{MARL C,J} \]
\[ \text{GETTAB C \ GET \ PROGRAM \ NAME \ OF \ SPECIFIED \ JOB} \]
\[ \text{JST (Q) \ RETURN \ TO \ PRL \ SUBPROGRAM} \]
\[ \text{SAME C,[LPTSLP1]} \]
\[ \text{JST CREDN2 \ IF \ NOT \ THIS \ JOB, \ TRY \ ANOTHER} \]
\[ \text{MOVC D,B} \]
\[ \text{LEVEL C,B \ WAKE \ UP \ SPOOLER} \]
\[ \text{JST (Q) \ GIVE \ UP} \]
\[ \text{NOSYN} \]
\[ \text{END} \]

\[ \text{NO \ ERRORS \ DETECTED} \]

\[ \text{PROGRAM \ BREAK \ IS \ 000250} \]

\[ \text{3K \ CORE \ USED} \]
TITLE AUX3

COMMENT %

WRITTEN BY GARRETT A. VANDE LUGT    COMPLETED 5-FEB-74

PART OF LRPRT, SHR, VERSION 3

AUX3.MAC CONTAINS A NUMBER OF SUBROUTINES WHICH ARE USED BY
PROGRAMS IN THE LRPRT 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

ENTRY = FROM LRPRT.MAC, LOCATED IN SECTION A

ENTRY = FROM RPRT.MAC, LOCATED IN SECTION C

ENTRY = FROM RPRT.MAC, LOCATED IN SECTION C
INCH - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION B
INRDO - FROM RPRTRP,MAC, LOCATED IN SECTION G
I00 - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION D
OOTA - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION J
OTTIM - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION K
OUTCHR - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION F
OUTDAT - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION J
OUTLES - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION I
OUTPPN - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION L
OUTSTR - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION E
OTTIM - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION K
OUTW - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION M
OUTWRO - FROM RPRTRP,MAC, LOCATED IN SECTION N
RTJUST - FROM RPRTRP,MAC AND LPTRPRP,MAC, LOCATED IN SECTION N

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN RPTLOW,MAC)
IBUT - BUFFER HEADER FOR DISK INPUT (3 WORDS)
IBUT2 - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)
OBUT - BUFFER HEADER FOR DISK OUTPUT (3 WORDS)
OBUT2 - 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 (10 WORDS)

MISEG
ENTRY INCHR, I0D, OUTCHR, INRDO, OUTW, II, OUTSTR, II, COLIN
INTERN OUTLES, OOTA, OUTDAT, OOTTIM, OUTPPN, OUTTH, RTJUST
EXTERN IBUT, PDLST, OBUT, IBUT2, OBUT2, PC2
SECTION A. THIS SUBROUTINE INPUTS CHARACTERS FROM A USER'S TERMINAL UNTIL AN END OF LINE IS ENCOUNTERED.

TTCALL 4.A INPUT NEXT CHARACTER
EOLINI CALL A.D IEND OF LINE
JIRST +2 INC. GO INPUT NEXT CHARACTER
JIRST (Q) IYES, RETURN

SECTION B. THIS SUBROUTINE INPUTS A CHARACTER FROM THE DISK CHANNEL 1. THE CHARACTER IS PLACED IN AC-A.

INCHR. SOSGE IBUF=2 IF DECREMENT CHARACTER COUNT
JIRST GETBUFF IF BUFFER IS EMPTY, GO GET ANOTHER BUFFER
GETNXT ILDS A,IBUF=1 IPUT CHARACTER IN AC-A
JIRST (Q) IRETURN
GETBUFF IN 1, IF FULL BUFFER
JIRST INCHR IF NO ERROR, GO GET NEXT CHARACTER
STATE 1,20000 IEND OF FILE?
JIRST (Q) IYES, RETURN
TTCALL 3,5AS71 DISK INPUT ERROR ON CHANNEL 1 + AUX3
HALT

SECTION C. THIS SUBROUTINE INPUTS AN INTERGER FROM A USER'S TERMINAL AND PLACES IT IN AC-B. IF THE 112 ENTRY POINT IS USED, LEADING SPACES ARE IGNORED.

SETZ B.
TTCALL 4.A INPUT CHARACTER
CALL A.60
CALE A.71
JIRST (Q) RETURN IF CHARACTER IS NOT A DIGIT
MULI B.12 MULTIPLY BY 10
ADDI B,-60(A) IADD VALUE OF DIGIT TO NO. IN AC-B
JIRST II+1 GO GET NEXT DIGIT
TTCALL 4.A INPUT CHARACTER
CAIN A.49 IS CHARACTER A SPACE?
JIRST +2 IYES, GO INPUT NEXT CHARACTER
REPRODUCED WITH PERMISSION OF THE COPYRIGHT OWNER. FURTHER REPRODUCTION PROHIBITED WITHOUT PERMISSION.

SECTION H. THIS SUBROUTINE OUTPUTS A DATA WORD TO THE DISK ON CHANNEL 4. THE DATA MUST BE IN AC-A.

SECTION I. THIS SUBROUTINE OUTPUTS LESSON NAMES TO THE DISK. THE LESSON NAME MUST BE IN AC-D.

SECTION J. THIS SUBROUTINE OUTPUTS A DATE TO THE DISK. IF THE DATE ENTRY POINT IS USED, THE DATE MUST BE IN AC-A; IF THE OUTDAT ENTRY POINT IS USED, THE CURRENT DATE IS OUTPUT.

SECTION K. THIS SUBROUTINE OUTPUTS A DATA WORD TO THE DISK.
AUX3

MACRO 47(284)-7 08145 2-AUG-74 PAGE 1-5

266 00121 047 01 0 000014
DATE A, I GET THE CURRENT DATE
267 00122 231 01 0 000037
IDIVI A,37 I DIVIDE TO GET DAY-1
268 00123 206 06 0 000002
MOVE F,B I SAVE IT IN AC-F
269 00124 350 30 0 000026
AOS F, I ADD ONE TO GET DAY
270 00125 231 01 0 000014
IDIVI A,14 I DIVIDE TO GET MONTH-1
271 00126 350 30 0 000002
AOS B I ADD ONE TO GET MONTH
272 00127 271 01 0 000010
ADDI A,120 I ADD 120 TO GET THE YEAR
273 00128 289 07 0 000001
MOVE G,A I SAVE THE YEAR IN AC-G
274 00129 201 01 0 000040
MOVE A,E I MOVE TO AC-E
275 00130 305 02 0 000012
CAI B,12 I CAI B,12
276 00131 265 17 0 000054
JSP Q,OUTCHR I OUTPUT SPACE IF MONTH<=18
277 00132 305 06 0 000012
CAI B,12 I CAI B,12
276 00133 265 17 0 000054
JSP Q,OUTCHR I OUTPUT SPACE IF DAY<=18
279 00134 201 01 0 000022
MOVE A,B I MOVE TO AC-B
280 00135 265 17 0 000033
JSP Q,10D I OUTPUT MONTH
281 00136 201 01 0 000057
MOVE A,"/" I MOVE A,"/"
282 00137 265 17 0 000054
JSP Q,OUTCHR I OUTPUT "/"
283 00138 202 01 0 000066
MOVE A,F I MOVE A,F
284 00139 265 17 0 000054
JSP Q,10D I OUTPUT DAY
285 00140 201 01 0 000057
MOVE A,"/" I MOVE A,"/"
286 00141 265 17 0 000054
JSP Q,OUTCHR I OUTPUT "/"
287 00142 204 31 0 000033
MOVE A,G I MOVE A,G
288 00143 265 17 0 000033
JSP Q,10D I OUTPUT YEAR
289 00144 202 01 0 000028
JRST #PC2 I RETURN

I SECTION K. THIS SUBROUTINE OUTPUTS THE TIME OF DAY TO THE DISK.
IF THE OUTTIME ENTRY POINT IS USED, THE TIME MUST BE IN AC-A. IF
THE OUTTIME ENTRY POINT IS USED, THE CURRENT DATE IS OUTPUT.

290 00215 202 17 0 000050
OTTM1HB MOVEQ PC,AC I SAVE RETURN ADDRESS
291 00216 254 00 0 000151
JRT OUTTM1HB I OUTPUT HOUR
292 00217 202 17 0 000051
OTTMIN MOVEQ AC,PC I SAVE RETURN ADDRESS
293 00218 407 01 0 000023
JRT OUTTMIN I MOVE CURRENT TIME OF DAY
294 00219 231 01 0 0016140
IDIVI A,1260000 I DIVIDE TO GET MINUTES SINCE MIDNIGHT
295 00220 231 01 0 000074
IDIVI A,1260 I DIVIDE TO GET HOURS SINCE MIDNIGHT
296 00221 254 00 0 000022
MOVE F,B I MOVE B,E
297 00222 254 00 0 000001
MOVE A,b I MOVE A,b
298 00223 305 02 0 000012
CAI B,018 I CAI B,018
299 00224 265 17 0 000041
JSP Q,OUTCHR I OUTPUT "0", IF HOURS<=10
300 00225 209 01 0 000002
MOVE A,B I MOVE A,B
301 00226 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
302 00227 201 01 0 000012
MOVE A,"/" I MOVE A,"/"
303 00228 265 17 0 000041
JSP Q,OUTCHR I OUTPUT "/", IF MINUTES<=10
304 00229 305 06 0 000012
CAI B,12 I CAI B,12
305 00230 265 17 0 000033
JSP Q,OUTCHR I OUTPUT "0", IF MINUTES<=10
306 00231 202 01 0 000030
MOVE A,F I MOVE A,F
307 00232 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
308 00233 305 06 0 000012
CAI B,12 I CAI B,12
309 00234 265 17 0 000033
JSP Q,OUTCHR I OUTPUT "0", IF MINUTES<=10
310 00235 202 01 0 000030
MOVE A,F I MOVE A,F
311 00236 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
312 00237 305 06 0 000012
CAI B,12 I CAI B,12
313 00238 265 17 0 000033
JSP Q,OUTCHR I OUTPUT "0", IF MINUTES<=10
314 00239 202 01 0 000030
MOVE A,F I MOVE A,F
315 00240 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
316 00241 305 06 0 000012
CAI B,12 I CAI B,12
317 00242 265 17 0 000033
JSP Q,OUTCHR I OUTPUT "0", IF MINUTES<=10
318 00243 202 01 0 000030
MOVE A,F I MOVE A,F
319 00244 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
320 00245 305 06 0 000012
CAI B,12 I CAI B,12
321 00246 265 17 0 000033
JSP Q,OUTCHR I OUTPUT "0", IF MINUTES<=10
322 00247 202 01 0 000030
MOVE A,F I MOVE A,F
323 00248 265 17 0 000033
JSP Q,10D I OUTPUT "0", IF MINUTES<=10
324 00249 128 74 0 000001
JRT #PC2 I RETURN
SECTION L. THIS SUBROUTINE OUTPUTS PROGRAMMER NUMBERS TO THE
IDISK. THE NUMBER MUST BE IN AC-A.

OUTPNI MOVE M,PC2  SSAVE RETURN ADDRESS
HRRE D,POLE81 SET UP PUSH DOWN POINTER IN AC-D
IDIVI A,10  IDIVIDE BY 10 TO GET A DIGIT
XRRI B,60 ICONVERT IT TO AN ASCII CHARACTER
PUSH D,B  INPUT CHARACTER IN PUSH DOWN LIST
PUSH D,C  NEXT DIGIT IF THERE IS ONE
FOPD A,1 IGET CHARACTER FROM PUSH DOWN LIST
TLNE D,.777777 IANY CHARACTERS LEFT?
JSTI ;+3 IYES, GO GET NEXT CHARACTER
JSTI PC2  IRETURN

SECTION M. THIS SUBROUTINE OUTPUTS LESSON TIMES AND ANWSERING
TIMES TO THE DISK. THE TIME MUST BE IN AC-A.

OUTTMI MOVE M,PC2  SSAVE RETURN ADDRESS
IDIVI A,D60000 IDIVIDE TO GET MINUTES
CALL B,D3000  ISHOW HOUR TIMES, IF 1/2 MIN. OR MORE IS LEFT
SKEP C
JSTP C,RTJUST  IRIGHT JUSTIFY IF AC-C DOES NOT CONTAIN ZERO
JSTP O,IOD  IOUTPUT TIME IN MINUTES
JSTI PC2  IRETURN

SECTION N. THIS SUBROUTINE RIGHT JUSTIFIES NUMBERS WHICH ARE
PRINTED IN COLUMNS ON LEARN REPORTS.

RTJUSTI MOVE E,2  SSAVE RETURN ADDRESS IN AC-E
MOVE P,AC  IPUT NO. IN AC-C
MOVE C,A  IPUT SPACE IN AC-A
CAIGE C,D10000
JSTP Q,OUTCHR  IOUTPUT SPACE IF NO. < 10000
CAIGE C,D1000
JSTP Q,OUTCHR  IOUTPUT SPACE IF NO. < 1000
CAIGE C,D10
JSTP Q,OUTCHR  IOUTPUT SPACE IF NO. < 10
MOVE A,C  IPUT NO. BACK IN AC-A
JSTI E  IRETURN
NOSYM  END
RPTLOW MACRO 47(204)-7 08147 2-AUG-74 PAGE 1

TITLE RPTLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUST COMPLETED 6-FEB-74

PART OF LRPRTR,SHR, VERSION 3

RPTLOW,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 LRPRTR 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,ONAM2,IBUFF2,OBUFF2,PARAM,ARGMT,FRAME
INTERN INBUFF,OUTBUFF,TRBLK,JOBN0,1,LIM1,LIM2
INTERN POLST,DATA,EXTOT,TMOT,ANSTOT,RESTOT,IMAC
INTERN LPTOT,INPUT,OUTPUT,DEVM,PPN,MONTH,DAY,ABD
INTERN INAM,ONAM,IBUFF,OBUFF,PC1,PC2,STORE,PPN,CMDLST,STOT

000137 JQ2VER.137
000137 LOC JQ2VER
00331 00001
000801 REL OC B

STORE: BLOCK 1 USED BY LRPRTR,MAC
ARGMT: BLOCK 4 USED BY LRPRTR,MAC
JOBN0: BLOCK 1 JOB NUMBER
TRBLK: BLOCK 3 USED BY LRPRTR,MAC (TRMOP, ARGUMENTS)
INTBLK: BLOCK 5 INTERRUPT BLOCK IS STORED HERE
INTLOC: BLOCK 12 INTERRUPT ROUTINE IS STORED HERE
IBUFF: BLOCK 3 USED BY RPRTR,MAC AND AUX,MAC
OBUFF: BLOCK 3 USED BY RPRTR,MAC AND AUX,MAC
INAM: BLOCK 4 USED BY RPRTR,MAC
ONAM: BLOCK 4 USED BY RPRTR,MAC AND RPRINT,MAC
PPNI: BLOCK 1 USER'S PROJECT-WRITER NO.
INBUFF: BLOCK 203 USED BY RPRTR,MAC AND LPRTR,MAC (CHANNEL 1 INPUT BUFFER)
OUTBUFF: BLOCK 203USED BY RPRTR,MAC AND LPRTR,MAC (CHANNEL 1 OUTPUT BUFFER)

ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN LOWL,MAC,
IPRSLOW,MAC, AND DLLOWL,MAC.

IBUFF: BLOCK 203 USED BY RPRTR,MAC (CHANNEL 3 INPUT BUFFER)
OBUFF: BLOCK 203 USED BY RPRTR,MAC (CHANNEL 4 OUTPUT BUFFER)
IBUFF2: BLOCK 3 USED BY RPRTR,MAC AND AUX3,MAC
OBUFF2: BLOCK 3 USED BY RPRTR,MAC AND AUX3,MAC
INAM2: BLOCK 4 USED BY RPRTR,MAC AND RPRINT,MAC
ONAM2: BLOCK 4 USED BY RPRTR,MAC AND RPRINT,MAC
PC1: BLOCK 1 USED BY RPRTR,MAC AND LPRTR,MAC
PC2: BLOCK 1 USED BY RPRTR,MAC AND AUX3,MAC
CMDLST: BLOCK 2 USED BY RPRTR,MAC AND RPRINT,MAC
<table>
<thead>
<tr>
<th>RPTLOW</th>
<th>MACRO</th>
<th>47(284)-7</th>
<th>00147</th>
<th>2-AUG-74</th>
<th>PAGE 1/1</th>
</tr>
</thead>
<tbody>
<tr>
<td>54</td>
<td>001291</td>
<td>POLSTI: BLOCK 12</td>
<td>IUSED BY RPRT, MAC AND AUX3, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>001271</td>
<td>LESSON: BLOCK 1</td>
<td>IUSED BY RPRT, MAC, LPRTRPT, MAC, AND RPRINT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>001221</td>
<td>DEVI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>001231</td>
<td>PPN1: BLOCK 3</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>001226</td>
<td>MONTY: BLOCK 1</td>
<td>IUSED BY RPRT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>001227</td>
<td>DAYTI: BLOCK 2</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>001131</td>
<td>ABOI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>001132</td>
<td>LIMI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>001133</td>
<td>FRAMA: BLOCK 2</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>001134</td>
<td>DATA: BLOCK 28</td>
<td>IUSED BY RPRT, MAC, LPRTRPT, MAC, AND RPRINT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>001136</td>
<td>STOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>001156</td>
<td>EXTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC, LPRTRPT, MAC, AND RPRINT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>001157</td>
<td>ANSTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>001161</td>
<td>RESTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>001162</td>
<td>RUNIT: BLOCK 1</td>
<td>IUSED BY RPRT, MAC, LPRTRPT, MAC, AND RPRINT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>001163</td>
<td>WUNITI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>001164</td>
<td>WTTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>001165</td>
<td>NTPTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>001166</td>
<td>YUPTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>001167</td>
<td>TUPPTOTI: BLOCK 1</td>
<td>IUSED BY RPRT, MAC AND LPRTRPT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>001170</td>
<td>PHAMTH: BLOCK 48</td>
<td>IUSED BY RPRT, MAC, LPRTRPT, MAC, AND RPRINT, MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>001170</td>
<td>NDSYM</td>
<td>END</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 001238

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. CFRAM,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 REENTRANT, BINARY CORE IMAGE OF THE LPROS
HIGH SEGMENT CAN BE WRITTEN ON THE DISK USING THE SAVE 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 EDIT 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

ARGMNT - ARGUMENTS FOR GETSEG UO (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

HISEG

EXTERN ARGMNT,PROS,PROS2,STORE

*002000* Flags = 0
*002001* A = 1
*002002* B = 2
*002017* QE = 7

!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 "GETS1", SINCE THE RETURN ADDRESS OF THE GETSEG UTO MUST BE THE SAME IN ALL HIGH SEGMENTS.

GETS1 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 ; GET THE COMMAND_INDEX IN AC-8
JRST 6, (B)
XWD B,TEST ; SEE SECTION F
XWD B,PRCS ; SEE SECTION C
XWD B,EP ; SEE SECTION D
XWD B,ET ; SEE SECTION G

SECTION C. JUMP TO PRCS TO PROCESS THE LESSON.

PRCS: TRZ FLAGS,2 ; CLEAR TEST COMMAND FLAG
JSP Q,PRCS
JRST NORUN ; SEE SECTION E

SECTION D. JUMP TO PRCS2 TO PROCESS THE LESSON.

PRCS2: TRZ FLAGS,2 ; CLEAR TEST COMMAND FLAG
JSP Q,PRCS2

SECTION E. SET UP COMMAND INDEX AND GETSEG ARGUMENTS BEFORE GETTING THE LEDIT HIGH SEGMENT.

NORUN: MOVE A,7 ; PUT COMMAND_INDEX IN AC-A
MOVE A,STORE ; SAVE COMMAND_INDEX IN STORE
MOVE A,ARGMT ; PUT ADDRESS OF GETSEG ARGUMENTS IN AC-A
MOVE B,ESI28 /LEDIT/ ; MOVE 'LEDIT' TO GETSEG ARGUMENTS
JRST GETS1 ; SEE SECTION A

SECTION F. JUMP TO PRCS TO PROCESS THE LESSON.

PRCS: TRZ FLAGS,2 ; SET TEST COMMAND FLAG
JSP Q,PRCS
JRST RUN ; SEE SECTION H

107 LPROS - MACRO '74(2844)-7 08149 2-AUG-74 PAGE 1-2
108 LPROS - MAC 6-JUN-74
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
SECTION G. JUMP TO PROC2 TO PROCESS THE LESSON.

SECTION H. SET UP COMMAND INDEX AND GETSEG ARGUMENTS BEFORE GETTING THE LEARN HIGH SEGMEN.

NO ERRORS DETECTED

PROGRAM BREAK IS 000042

3K CORE USED
TITLE PROS

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT     COMPLETED 16-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 UNSTRT 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
PRECEDE 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 CHARACTER 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 DELIN STATEMENTS.

SECTION K. PROCESSES MATCH STATEMENTS.

SECTION L. PROCESSES ANTICIPATED ANSWERS.

SECTION M. PROCESSES ACTION STATEMENTS.

SECTION N. PROCESSES ACTION ANSWER TAGS.
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Processes LI statements.</td>
</tr>
<tr>
<td>1</td>
<td>Processes FI and GI statements.</td>
</tr>
<tr>
<td>2</td>
<td>Processes AI statements.</td>
</tr>
<tr>
<td>3</td>
<td>Processes SI statements.</td>
</tr>
<tr>
<td>4</td>
<td>Processes RI statements.</td>
</tr>
<tr>
<td>5</td>
<td>Processes EI statements.</td>
</tr>
<tr>
<td>6</td>
<td>Processes GI statements.</td>
</tr>
<tr>
<td>7</td>
<td>Processes BI statements.</td>
</tr>
<tr>
<td>8</td>
<td>Processes CI statements.</td>
</tr>
<tr>
<td>9</td>
<td>Processes R-frames.</td>
</tr>
<tr>
<td>10</td>
<td>Processes B-frames.</td>
</tr>
<tr>
<td>11</td>
<td>Calls subroutine to process C-frames.</td>
</tr>
<tr>
<td>12</td>
<td>Makes second pass through lesson.</td>
</tr>
<tr>
<td>13</td>
<td>Makes third pass through lesson.</td>
</tr>
<tr>
<td>14</td>
<td>Closes processed lesson file and prints error summary.</td>
</tr>
</tbody>
</table>

**List of code characters produced by PROS.MAC**

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;02&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>GI</td>
</tr>
<tr>
<td>&lt;10&gt;</td>
<td>CI</td>
</tr>
<tr>
<td>&lt;11&gt;</td>
<td>Extra</td>
</tr>
<tr>
<td>&lt;13&gt;</td>
<td>Order</td>
</tr>
<tr>
<td>&lt;14&gt;</td>
<td>Phonetic</td>
</tr>
<tr>
<td>&lt;16&gt;</td>
<td>Page</td>
</tr>
<tr>
<td>&lt;17&gt;</td>
<td>Calc</td>
</tr>
<tr>
<td>&lt;23&gt;</td>
<td>Delim</td>
</tr>
<tr>
<td>&lt;24&gt;</td>
<td>Record</td>
</tr>
<tr>
<td>&lt;26&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 CFRAK.MAC (SECTION A)

CFRAK = PROCESSES C-FRAMES, LOCATED IN CFRAK.MAC (SECTION A)

CSTATE = PROCESSES C-FRAME STATEMENT FOLLOWING A CI, LOCATED IN CFRAK.MAC (SECTION A)

FRM1N = TYPES FRAME AND LINE NOS. IN AN ERROR MESSAGE AND
SCANS TO END OF LINE, LOCATED IN AUX4.MAC (SECTION F)

FRM1N2 = 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)

I1T = 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)

I0D = OUTPUTS AN INTEGER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION D)
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>IOT - OUTPUTS AN INTEGER IN AC-A TO A USER'S TERMINAL, LOCATED IN AUX4.MAC (SECTION E)</td>
</tr>
<tr>
<td>161</td>
<td></td>
</tr>
<tr>
<td>162</td>
<td></td>
</tr>
<tr>
<td>163</td>
<td>OUTCHR - OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION B)</td>
</tr>
<tr>
<td>164</td>
<td></td>
</tr>
<tr>
<td>165</td>
<td></td>
</tr>
<tr>
<td>166</td>
<td>TXTYP - PROCESSES TEXT VARIABLES, LOCATED IN CFRAM.MAC (SECTION A)</td>
</tr>
<tr>
<td>167</td>
<td></td>
</tr>
<tr>
<td>168</td>
<td>TYSYM - TYPES THE SYMBOL IN AC-G, LOCATED IN AUX4.MAC (SECTION G)</td>
</tr>
<tr>
<td>169</td>
<td></td>
</tr>
<tr>
<td>170</td>
<td></td>
</tr>
<tr>
<td>171</td>
<td>EXTERNAL MEMORY REFERENCES (ALL LOCATED IN PRSLow.MAC)</td>
</tr>
<tr>
<td>172</td>
<td></td>
</tr>
<tr>
<td>173</td>
<td>ADDR - ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO VARIABLES APPEARING IN G-FRAME STATEMENTS</td>
</tr>
<tr>
<td>174</td>
<td></td>
</tr>
<tr>
<td>175</td>
<td>DELS - LIST OF WORD DELIMITERS (17 WORDS)</td>
</tr>
<tr>
<td>176</td>
<td></td>
</tr>
<tr>
<td>177</td>
<td>ERRTOT - TOTAL NO. OF ERRORS DETECTED</td>
</tr>
<tr>
<td>178</td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>FRHNUM - CURRENT FRAME NO.</td>
</tr>
<tr>
<td>180</td>
<td></td>
</tr>
<tr>
<td>181</td>
<td>FRHTOT - TOTAL NO. OF FRAMES</td>
</tr>
<tr>
<td>182</td>
<td></td>
</tr>
<tr>
<td>183</td>
<td>FRMTYP - CURRENT FRAME TYPE</td>
</tr>
<tr>
<td>184</td>
<td></td>
</tr>
<tr>
<td>185</td>
<td>INAM - DISK LOOKUP ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>186</td>
<td></td>
</tr>
<tr>
<td>187</td>
<td>LINE - CURRENT LINE NO.</td>
</tr>
<tr>
<td>188</td>
<td></td>
</tr>
<tr>
<td>189</td>
<td>NUMIST - BEGINNING OF CHARACTER COUNT LIST</td>
</tr>
<tr>
<td>190</td>
<td></td>
</tr>
<tr>
<td>191</td>
<td>ONAM - DISK ENTER ARGUMENTS (4 WORDS)</td>
</tr>
<tr>
<td>192</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>PART - G-FRAME PART NO.</td>
</tr>
<tr>
<td>194</td>
<td></td>
</tr>
<tr>
<td>195</td>
<td>PC1 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>196</td>
<td></td>
</tr>
<tr>
<td>197</td>
<td>PC2 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>198</td>
<td></td>
</tr>
<tr>
<td>199</td>
<td>PC3 - SUBROUTINES STORE RETURN ADDRESS HERE</td>
</tr>
<tr>
<td>200</td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>PRSFN - NO. OF LAST FRAME TO BE PROCESSED</td>
</tr>
<tr>
<td>202</td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>PRSTT - NO. OF FIRST FRAME TO BE PROCESSED</td>
</tr>
<tr>
<td>204</td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>RFMCNT - NO. OF REVIEW FRAMES IN THE LESSON</td>
</tr>
<tr>
<td>206</td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>RSVNUM - TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE</td>
</tr>
<tr>
<td>208</td>
<td></td>
</tr>
<tr>
<td>209</td>
<td>SYMIDX - INDEX TO SYMBOL TABLE</td>
</tr>
<tr>
<td>210</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td></td>
</tr>
<tr>
<td>212</td>
<td></td>
</tr>
</tbody>
</table>
319 002073 386 21 0 0 008040  CAIN A. 40 IS CHARACTER A SPACE?
320 002074 254 46 0 0 008072  JRST +2 YES, GO INPUT NEXT CHARACTER
321 002075 302 21 0 0 002155  JRST A.15 END OF COMMAND STRING?
322 002076 254 20 0 0 003544  JRST ILLSHT NO, GO TYPE ERROR MESSAGE
323 002077 222 22 0 0 002155  MOVE B, PRSFN STORE NO. OF LAST FRAME TO BE PROCESSED
324 002126 254 22 0 0 002116  JRST GETLES
325 002121 202 32 0 0 003967  ONEUNII MOVE B, PRSFN STORE NO. OF 1ST FRAME TO BE PROCESSED
326 002122 202 32 0 0 002167  JRST GETLES
327 002123 254 00 0 0 002116  JRST GETLES
328 002124 305 21 0 0 002155  NAMINI CAIN A.15 END OF COMMAND STRING?
329 002125 254 26 0 0 002116  JRST GETLES YES, GO TO GETLES
330 002126 254 29 0 0 003110  JRST +2
331 002127 251 34 1 0 007021  NAMINI1 TDCALL 4,A INPUT NEXT CHARACTER
332 002128 305 01 0 0 002155  CAIN A. 30 IS CHARACTER A SPACE?
333 002129 254 20 0 0 003110  JRST +2 YES, GO INPUT NEXT CHARACTER
334 002130 305 21 0 0 002155  CAIN A. "/" IS CHARACTER A "/"?
335 002131 254 76 0 0 002155  JRST GETSHT YES, GO TO GETSHT
336 002132 302 21 0 0 002155  CAIN A. 15 END OF COMMAND STRING?
337 002133 254 26 0 0 002155  JRST ILLSHT NO, GO TYPE ERROR MESSAGE
338 002134 305 21 0 0 002155  NAMINI CAIN A.15 WAS LAST CHARACTER A RETURN?
339 002135 251 04 0 0 002155  TDCALL 4,A YES, INPUT LINE FEED
340 002136 202 37 0 0 002101  SET G, 000101
341 002137 202 31 0 0 003263  MOVIS A, IPRS1 MOVE EXTENSION TO LOOKUP ARGUMENTS
342 002138 305 01 0 0 002121  MOVE A, INAM+1 MOVEL EXTENSION TO LOOKUP ARGUMENTS
343 002139 202 20 0 0 002033  SETEM INAM=3
344 002140 202 26 0 0 002166  MOVS A, INAM=1 LOOK FOR PROCESSED LESSON FILE
345 002141 202 47 0 0 002166  SETO G, SET AG TO ONES IF NOT FOUND
346 002142 139 04 2 0 002211  LDB D, TIPMT GET CREATION TIME IN AC-D
347 002143 139 03 2 0 002021  LDB C, DATPNT GET CREATION DATE IN AC-C
348 002144 202 01 0 0 005560  MOVIS A, UNP1 MOVE A, INAM=1 MOVE EXTENSION TO LOOKUP ARGUMENTS
349 002145 202 31 0 0 002166  SETEM INAM=3
350 002146 202 30 0 0 002033  MOVS A, INAM=1 LOOKUP UNPROCESSED LESSON FILE
351 002147 202 29 0 0 002166  JUMP G, PROC1 IF NOT FOUND, GO TYPE ERROR MESSAGE
352 002148 326 27 0 0 002147  IF NO PROCESSED FILE, GO TO PROC1
353 002149 139 22 0 0 002147  JUMP B, TIPMT GET CREATION TIME IN AC-B
354 00214A 139 01 0 0 002145  LDB A, DATPNT GET CREATION DATE IN AC-A
355 00214B 315 01 0 0 002145  CANN A,C ARE THE DATES EQUAL?
356 00214C 202 27 0 0 002166  JSR A=4, 2 IF YES, SKIP TO TIME CHECK
357 00214D 254 30 1 0 002166  CAMG A,C UNPROCESSED FILE OLDER THAN PROCESSED FILE?
358 00214E 254 26 0 0 002166  JRST #PC1 YES, RETURN TO LPROS.MAC (NO NEED TO PROCESS)
359 00214F 254 20 0 0 002147  JRST #PC1 YES, RETURN TO LPROS.MAC (NO NEED TO PROCESS)
360 002150 317 02 0 0 002166  CAMG B,D UNPROCESSED FILE OLDER THAN PROCESSED FILE?
361 002151 254 30 1 0 002166  JRST #PC1 YES, RETURN TO LPROS.MAC (NO NEED TO PROCESS)
362 002152 203 13 0 0 002133  MOVE A, INAM=1 MOVE A, INAM=1
363 002153 202 31 0 0 002166  MOVEM A, ONAME IMOVE LESSON NAME TO ENTER ARGUMENTS
364 002154 205 31 0 0 004560  MOVIS A, IPRS1 IMOVE EXTENSION TO ENTER ARGUMENTS
365 002155 205 31 0 0 002166  MOVEM A, ONAME=1 IMOVE EXTENSION TO ENTER ARGUMENTS
366 002156 202 01 0 0 002166  SETM ONAME=2
367 002157 402 00 0 0 002166  SETM ONAME=3
368 002158 207 32 0 0 002166  ENTER 2, ONAME IMOVE EXTENSION TO ENTER ARGUMENTS
369 002159 254 28 8 0 002167  JRST NORDOM IF DISK QUOTA EXCEEDED, GO TYPE ERROR MESSAGE
370 00215A 254 28 8 0 002167  JRST PROC SEE SECTION B
SECTION B. INITIALIZE THE LESSON PROCESSOR.

PROC1  SETERR, TOT ;SET ERROR COUNT AND CHARACTER COUNT TO ZERO

PROC1  MOVEM  A, 4 ;MOVE A TO TOT

PROC1  MOVEM  A, 5 ;MOVE A TO SYM

PROC1  MOVEM  A, 6 ;MOVE A TO HT

PROC1  MOVEM  A, 7 ;MOVE A TO SYM

PROC1  MOVEM  A, 8 ;MOVE A TO HT

PROC1  SETHFM, 1 ;SETH FRAME COUNT TO ZERO

PROC1  SETFMTOT, 1 ;SET FRAME TOTAL COUNT TO ZERO

PROC1  SETSYM, 1 ;SET SYMBOL TABLE INDEX TO ZERO

PROC1  SETSYM, 1 ;SET SYMBOL TABLE INDEX TO ZERO

PROC1  SETHFTAB, 1 ;SET 1ST WORD IN SYMBOL TABLE TO ZERO

PROC1  SETHFRMCNT, 1 ;SET REVFRAME COUNT TO ZERO

PROC1  SETUP, A, 60 ;SET UP LIST OF STANDARD WORD DELIMITERS

PROC1  MOVEA  60 ;MOVE A TO DELS

PROC1  MOVEM  A, 4 ;MOVE A TO HT

PROC1  MOVEM  A, 1 ;MOVE A TO HT

PROC1  MOVEM  A, 2 ;MOVE A TO HT

PROC1  MOVEM  A, 3 ;MOVE A TO HT

PROC1  MOVEA  64 ;MOVE A TO DELS

PROC1  MOVEM  A, 11 ;MOVE A TO HT

PROC1  MOVEM  A, 12 ;MOVE A TO HT

PROC1  MOVEM  A, 13 ;MOVE A TO HT

PROC1  MOVEM  A, 14 ;MOVE A TO HT

PROC1  MOVEM  A, 15 ;MOVE A TO HT

PROC1  MOVEM  A, 16 ;MOVE A TO HT

PROC1  SETHFM, 7 ;TERMINATE LIST WITH ZERO

PROC1  GFSFTM, 1 ;INPUT CHARACTER

PROC1  CAI N, 4 ;END OF LESSON?

PROC1  J RST  NOPR ;YES, GO TO NOPR

PROC1  CAI N, 36 ;BEGINNING OF FRAME?

PROC1  J RST  GFSFTM ;NO, GO NEXT CHARACTER

PROC1  J RST  Q, 11 ;INPUT FRAME NO.
<table>
<thead>
<tr>
<th>PROG</th>
<th>MACRO 47(234)</th>
<th>081050</th>
<th>2-AUG-74 PAGE 1-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>425</td>
<td>02231</td>
<td>315 02 0</td>
<td>00 020100</td>
</tr>
<tr>
<td>426</td>
<td>02232</td>
<td>254 20 0</td>
<td>00 020202</td>
</tr>
<tr>
<td>427</td>
<td>02233</td>
<td>254 03 0</td>
<td>00 022431</td>
</tr>
<tr>
<td>428</td>
<td>02234</td>
<td>251 11 0</td>
<td>00 000000</td>
</tr>
<tr>
<td>429</td>
<td>TDCALL 3: CASCIF /</td>
<td></td>
<td></td>
</tr>
<tr>
<td>430</td>
<td>NO FRAMES WERE PROCESSED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>431</td>
<td>02235</td>
<td>303 03 0</td>
<td>00 022511</td>
</tr>
<tr>
<td>432</td>
<td>02236</td>
<td>669 00 0</td>
<td>00 000000</td>
</tr>
</tbody>
</table>

**SECTION C. PROCESS THE FIRST LINE IN A FRAME.**

<table>
<thead>
<tr>
<th>PROG</th>
<th>MACRO 47(234)</th>
<th>081050</th>
<th>2-AUG-74 PAGE 1-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>436</td>
<td>02237</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>437</td>
<td>02238</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>438</td>
<td>02239</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>439</td>
<td>02240</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>440</td>
<td>02241</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>441</td>
<td>02242</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>442</td>
<td>02243</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>443</td>
<td>02244</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>444</td>
<td>02245</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>445</td>
<td>02246</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>446</td>
<td>02247</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>447</td>
<td>02248</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>448</td>
<td>02249</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>449</td>
<td>02250</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>450</td>
<td>02251</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>451</td>
<td>02252</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>452</td>
<td>02253</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>453</td>
<td>02254</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>454</td>
<td>02255</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>455</td>
<td>02256</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>456</td>
<td>02257</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>457</td>
<td>02258</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>458</td>
<td>02259</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>459</td>
<td>02260</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>460</td>
<td>02261</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>461</td>
<td>02262</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>462</td>
<td>02263</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>463</td>
<td>02264</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>464</td>
<td>02265</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>465</td>
<td>02266</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>466</td>
<td>02267</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>467</td>
<td>02268</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>468</td>
<td>02269</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
<tr>
<td>469</td>
<td>02270</td>
<td>023 17 0</td>
<td>00 02230</td>
</tr>
</tbody>
</table>

**SECTION D. PROCESS THE FIRST WORD IN A Q-FRAME STATEMENT.**

<table>
<thead>
<tr>
<th>PROG</th>
<th>MACRO 47(234)</th>
<th>081050</th>
<th>2-AUG-74 PAGE 1-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>470</td>
<td>02271</td>
<td>355 00 0</td>
<td>00 000266</td>
</tr>
<tr>
<td>471</td>
<td>02272</td>
<td>005 03 0</td>
<td>00 000000</td>
</tr>
<tr>
<td>472</td>
<td>02273</td>
<td>005 03 0</td>
<td>00 000000</td>
</tr>
<tr>
<td>473</td>
<td>02274</td>
<td>005 03 0</td>
<td>00 000000</td>
</tr>
</tbody>
</table>
SECTION E. PROCESS A TEXT STATEMENT.

TEXT1 MOVE B,PAR T ;GET PART NO. IN AC-B

CAIE 8.1 ;IS PART NO. <= 17?

JST SQERR1 ;NO, GO TYPE ERROR MESSAGE

MOVE B,A1 ;SAVE CHARACTER IN AC-A

RECOVI MOVE 1, A, 1

MOVEM A,PAR T ;SET PART NO. TO 1

JST QOUTCHR ;OUTPUT CHARACTE R

JST U, 2 ;IND.JUMP

MOVE A. B ;GET CHARACTER BACK IN AC-A

TEXT2 CAIN A, "S" ;IS CHARACTER A "S"?

JST TT1 ;YES, GO TO TT1

CAIE A,15 ;IS CHARACTER A RETURN?

JST U, 2 ;IND.JUMP

JST QOUTCHR ;OUTPUT THE RETURN CHARACTER

MOVE A, 1, 2 ;OUTPUT CHARACTER

ACTXT1 JSP Q,OUTCHR ;OUTPUT CHARACTER

TT1 JSP Q, INCHR ;INPUT CHARACTER

TT1 JSP Q, OUTCHR ;OUTPUT CHARACTER

JSP Q, INCHR ;INPUT CHARACTER

CAIE A, "\" ;IS CHARACTER A SLASH?

JST SLASH ;YES, GO TO SLASH

CAIE A,4 ;END OF LESSON?

CAIE A,16 ;END OF FRAME?

JST ERR4 ;YES, GO TYPE ERROR MESSAGE

CAIE A,12 ;IS CHARACTER A LINE FEED?

JST TT ;IND.GOTO OUTPUT CHARACTER

SOERR1 MOVE E, 0, 39 ;SAVE CHARACTER IN AC-A

JSP Q, FRMLN2 ;TYPE FRAME AND LINE NOS.

TT CALL 3, [ASCZC] /
<table>
<thead>
<tr>
<th>PROS</th>
<th>MACRO</th>
<th>47(264)-7 0910-2</th>
<th>2-AUG-74</th>
<th>PAGE 1-14</th>
</tr>
</thead>
<tbody>
<tr>
<td>743</td>
<td>022671</td>
<td>254 0 0 0 0 026711</td>
<td>JRST PGERR1</td>
<td>IO, GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>744</td>
<td>022672</td>
<td>202 0 0 0 0 0026720</td>
<td>MOVE B,PART</td>
<td>SET PART NO. TO 1</td>
</tr>
<tr>
<td>745</td>
<td>022673</td>
<td>202 0 0 0 0 027001</td>
<td>MOVE B,A</td>
<td>SAVE THE CHARACTER IN AC-A</td>
</tr>
<tr>
<td>746</td>
<td>022674</td>
<td>201 0 0 0 0 003106</td>
<td>MOVE V,A,16</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>747</td>
<td>022675</td>
<td>202 0 0 0 0 002701</td>
<td>JSF G,OUTCHR</td>
<td>OUTPUT CHARACTER BACK IN AC-A</td>
</tr>
<tr>
<td>748</td>
<td>022676</td>
<td>202 0 0 0 0 003102</td>
<td>JRST +2</td>
<td>INPUT CHARACTER</td>
</tr>
<tr>
<td>750</td>
<td>022700</td>
<td>265 17 0 0 002700</td>
<td>JSF G,INCHR</td>
<td>INPUT CHARACTER BACK IN AC-A</td>
</tr>
<tr>
<td>751</td>
<td>022701</td>
<td>302 0 0 0 0 002701</td>
<td>CAIE A,11</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>752</td>
<td>022702</td>
<td>326 0 0 0 0 002700</td>
<td>CAIE A,15</td>
<td>IS CHARACTER A RETURN?</td>
</tr>
<tr>
<td>753</td>
<td>022703</td>
<td>254 0 0 0 0 002701</td>
<td>JSF G,INCHR</td>
<td>INPUT THE RETURN</td>
</tr>
<tr>
<td>754</td>
<td>022704</td>
<td>326 0 0 0 0 003001</td>
<td>JSF G,INCHR</td>
<td>INPUT LINE FEED</td>
</tr>
<tr>
<td>755</td>
<td>022705</td>
<td>254 0 0 0 0 002701</td>
<td>JRST -2</td>
<td>IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER</td>
</tr>
<tr>
<td>756</td>
<td>022706</td>
<td>265 17 0 0 002700</td>
<td>JRST DFRAKE</td>
<td>GO TO DFRAKE (SECTION D)</td>
</tr>
<tr>
<td>759</td>
<td>022711</td>
<td>265 17 0 0 002664</td>
<td>JRST DFRAKE</td>
<td>GO TO DFRAKE (SECTION D)</td>
</tr>
<tr>
<td>760</td>
<td>022712</td>
<td>051 3 0 0 0 026662</td>
<td>TCALL 3,ASCZ /</td>
<td></td>
</tr>
<tr>
<td>761</td>
<td>022713</td>
<td>254 0 0 0 0 022713</td>
<td>ILLEGAL</td>
<td>PAGE STATEMENT AFTER ANSWER OR ACTION STATEMENT/3</td>
</tr>
<tr>
<td>762</td>
<td>022714</td>
<td>265 17 0 0 002713</td>
<td>ILLEGAL</td>
<td>PAGE STATEMENT FORMAT/3</td>
</tr>
<tr>
<td>763</td>
<td>022715</td>
<td>254 0 0 0 0 002715</td>
<td>JRST DFRAKE</td>
<td>GO TO DFRAKE (SECTION D)</td>
</tr>
<tr>
<td>764</td>
<td>022716</td>
<td>265 17 0 0 002716</td>
<td>JRST DFRAKE</td>
<td>GO TO DFRAKE (SECTION D)</td>
</tr>
<tr>
<td>765</td>
<td>022717</td>
<td>281 21 0 0 022717</td>
<td>MOVE V,A,1</td>
<td>CAIE A,1PART</td>
</tr>
<tr>
<td>766</td>
<td>022718</td>
<td>315 21 0 0 002718</td>
<td>MOVE V,A,1</td>
<td>IS PART NO. &lt; 13</td>
</tr>
<tr>
<td>767</td>
<td>022719</td>
<td>254 0 0 0 0 001033</td>
<td>JRST SRERR2</td>
<td>GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>768</td>
<td>022720</td>
<td>254 0 0 0 0 001033</td>
<td>JRST SRERR2</td>
<td>GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>769</td>
<td>022721</td>
<td>254 0 0 0 0 001033</td>
<td>JRST SRERR2</td>
<td>GO TYPE ERROR MESSAGE</td>
</tr>
<tr>
<td>770</td>
<td>022722</td>
<td>202 0 0 0 0 007220</td>
<td>MOVE V,A,1</td>
<td>MOVE &lt;E2&gt; TO AC-A</td>
</tr>
<tr>
<td>771</td>
<td>022723</td>
<td>202 0 0 0 0 002273</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>772</td>
<td>022724</td>
<td>265 17 0 0 002724</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>773</td>
<td>022725</td>
<td>265 17 0 0 002725</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>774</td>
<td>022726</td>
<td>302 0 0 0 0 020011</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>775</td>
<td>022727</td>
<td>306 0 0 0 0 020010</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>776</td>
<td>022728</td>
<td>304 0 0 0 0 002728</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>777</td>
<td>022729</td>
<td>302 0 0 0 0 002729</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>778</td>
<td>022730</td>
<td>326 0 0 0 0 002730</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>779</td>
<td>022731</td>
<td>254 0 0 0 0 002731</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>780</td>
<td>022732</td>
<td>265 17 0 0 002732</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>781</td>
<td>022733</td>
<td>265 17 0 0 002733</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>782</td>
<td>022734</td>
<td>265 17 0 0 002734</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>783</td>
<td>022735</td>
<td>265 17 0 0 002735</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>784</td>
<td>022736</td>
<td>265 17 0 0 002736</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>785</td>
<td>022737</td>
<td>302 0 0 0 0 002737</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>786</td>
<td>022738</td>
<td>302 0 0 0 0 002738</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>787</td>
<td>022739</td>
<td>302 0 0 0 0 002739</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>788</td>
<td>022740</td>
<td>302 0 0 0 0 002740</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>789</td>
<td>022741</td>
<td>302 0 0 0 0 002741</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>790</td>
<td>022742</td>
<td>302 0 0 0 0 002742</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>791</td>
<td>022743</td>
<td>302 0 0 0 0 002743</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>792</td>
<td>022744</td>
<td>302 0 0 0 0 002744</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>793</td>
<td>022745</td>
<td>302 0 0 0 0 002745</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>794</td>
<td>022746</td>
<td>302 0 0 0 0 002746</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
<tr>
<td>795</td>
<td>022747</td>
<td>302 0 0 0 0 002747</td>
<td>JRST OFRAME</td>
<td>GO TO OFRAME (SECTION D)</td>
</tr>
</tbody>
</table>
ILLEGAL SLIDE STATEMENT FORMAT.  
JST QFRAME  ISEE SECTION D  

ERRR: JSP 0,FRMLN  ITYPE FRAME AND LINE NOS.  

TCALL 3,[ASCIZ /  

SLIDE NUMBER IS NOT IN THE RANGE 0 THRU 99/}  

JST QFRAME  ISEE SECTION D  

ERRR: JSP 0,FRMLN  ITYPE FRAME AND LINE NOS.  

TCALL 3,[ASCIZ /  

PROJECTION TIME EXCEEDS 90 SECONDS/}  

JST QFRAME  ISEE SECTION D  

<table>
<thead>
<tr>
<th>SECTIONS.</th>
<th>CALC, HINT, BOTH, STORE, REVIEW, OR RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVEI 9,2</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST CALCK</td>
<td>NO, GO TO CALCK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>CALC STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/}</td>
<td></td>
</tr>
<tr>
<td>CALCN: MOVEI 1,26</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST HINTK</td>
<td>NO, GO TO HINTK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>HINT STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/}</td>
<td></td>
</tr>
<tr>
<td>BOTH; MOVEI 9,2</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST BOTHK</td>
<td>NO, GO TO BOTHK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>BOTH STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/}</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST BOTHK</td>
<td>NO, GO TO BOTHK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>STORE; MOVEI 9,2</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST STOROK</td>
<td>NO, GO TO STOROK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>STORE STATEMENT DOES NOT PRECEDE ANSWER STATEMENT/}</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST REVDK</td>
<td>NO, GO TO REVDK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
<tr>
<td>REVIEW; MOVEI 9,2</td>
<td></td>
</tr>
<tr>
<td>CAMLE B,PART; IHAS AN ANSWER STATEMENT BEEN ENCONTRITED?</td>
<td></td>
</tr>
<tr>
<td>JST STOROK</td>
<td>NO, GO TO STOROK</td>
</tr>
<tr>
<td>JSP 0,FRMLN</td>
<td>ITYPE FRAME AND LINE NOS.</td>
</tr>
<tr>
<td>TCALL 3,[ASCIZ /</td>
<td></td>
</tr>
</tbody>
</table>
I: SECTION M. PROCESS AN ANSWER STATEMENT.

976 00121' 201 02 00 000002  ANSWER: MOVEV B,2
977 00123' 515 02 00 001113  CAMGE B,2,PART 1: DOES ANSWER PRECEDE ACTION?
978 00125' 254 02 00 001261  JSP Q,INCHR 1: NO, GO TYPE ERROR STATEMENT
979 00125' 316 02 00 001261  CANN B,2,PART 1: YES THIS THE 2ND ANSWER STATEMENT IN A FRAME?
980 00126' 254 02 00 001264  JSP Q,OUTCHR 1: YES, GO TYPE ERROR MESSAGE
981 00126' 202 02 00 001259  MOVEM B,2,PART 1: SET PART NO. TO 2
982 00127' 200 02 00 002001  RECOV3: MOVE B,A 1: SAVE CHARACTER IN AC-A
983 00127' 201 01 00 000203  MOVEV A,3
984 00132' 265 17 00 001163  JSP Q,OUTCHR 1: OUTPUT A <23>
987 00127' 203 01 00 000202  MOVE A,B 1: PUT CHARACTER BACK IN AC-A
988 00126' 306 01 00 000252  CANN A,2,PART 1: IS CHARACTER A "#97"
989 00122' 254 17 00 001222  JSP Q,OUTCHR 1: YES, OUTPUT CHARACTER
990 00126' 254 01 00 003262  CAIE A,2,PART 1: IS CHARACTER A "#97"?
991 00127' 254 02 00 001231  JSP Q,INCHR 1: INPUT NEXT INSTRUCTION
992 00126' 302 01 00 000211  CAIE A,11
993 00126' 302 01 00 000240  CAIE A,48
994 00125' 254 02 00 003215  JSP Q,OUTCHR 1: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
995 00125' 302 01 00 003215  JSP Q,INCHR 1: END OF ANSWER STATEMENT?
996 00125' 254 17 00 007265  JSP Q,INCHR 1: YES, GO TO NOTIME
997 00125' 254 08 00 001241  JSP Q,INCHR 1: INPUT WAIT TIME
998 00125' 254 17 00 001238  JSP Q,INCHR 1: INPUT CHARACTER
999 00124' 302 01 00 003011  CAIE A,11
1000 00124' 302 01 00 003640  CAIE A,48
1001 00123' 254 02 00 001240  JSP Q,INCHR 1: IF CHARACTER IS A SPACE OR A TAB, GO INPUT NEXT CHARACTER
1002 00124' 302 01 00 003025  JSP Q,INCHR 1: END OF ANSWER STATEMENT?
1003 00124' 254 02 00 003015  JSP Q,INCHR 1: NO, GO TYPE ERROR MESSAGE
1004 00124' 303 02 00 001764  JSP Q,INCHR 1: WAIT TIME LONGER THAN 8100 SECONDS?
1005 00124' 254 02 00 001267  JSP Q,INCHR 1: YES, GO TYPE ERROR MESSAGE
1006 00125' 254 02 00 008132  IDIVI B,2,990
1007 00125' 271 02 00 008004  ADDI B,40
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Microcode</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1008</td>
<td>001252'</td>
<td>271 33 0 0</td>
<td>0000040 ADDI C,40</td>
</tr>
<tr>
<td>1009</td>
<td>001253'</td>
<td>200 01 0 0</td>
<td>0000002 MOVE A,00002</td>
</tr>
<tr>
<td>1010</td>
<td>001254'</td>
<td>265 17 0 0</td>
<td>001225* JSP Q,OUTCHR :OUTPUT 1ST DIGIT OF TIME (BASE 90)</td>
</tr>
<tr>
<td>1011</td>
<td>001255'</td>
<td>203 01 0 0</td>
<td>0000033 MOVE A,C</td>
</tr>
<tr>
<td>1012</td>
<td>001256'</td>
<td>265 17 0 0</td>
<td>001254* JSP Q,OUTCHR :OUTPUT 2ND DIGIT OF TIME (BASE 90)</td>
</tr>
<tr>
<td>1013</td>
<td>001257'</td>
<td>200 11 0 0</td>
<td>0000015 MOVEI A.15</td>
</tr>
</tbody>
</table>
| 1014 | 001258' | 265 06 0 0 | 001272* NOTIME | PUNT 
| 1015 | 001259' | 265 17 0 0 | 001119* SQERR3 | JSP Q,FRMLN2 ;TYPE FRAME AND LINE NOS. |
| 1016 | 001260' | 051 33 0 0 | 003056* TTCALL 3,[ASCII] |
| 1017 | 001261' | 051 33 0 0 | 003056* ANSWER STATEMENT AFTER ACTION STATEMENT/ |
| 1018 | 001262' | 254 08 0 0 | 001226* JRST RECOV3 |
| 1019 | 001263' | 265 17 0 0 | 001261* SQERR4 | JSP Q,FRMLN2 ;TYPE FRAME AND LINE NOS. |
| 1020 | 001264' | 051 33 0 0 | 003067* TTCALL 3,[ASCII] |
| 1021 | 001265' | 051 33 0 0 | 003067* MORE THAN ONE ANSWER STATEMENT IN A SINGLE Q-FRAME/ |
| 1022 | 001266' | 254 08 0 0 | 001226* JRST RECOV3 |
| 1023 | 001267' | 265 17 0 0 | 001227* ERR12: | JSP Q,FRMLN ;TYPE FRAME AND LINE NOS. |
| 1024 | 001268' | 051 33 0 0 | 003123* TTCALL 3,[ASCII] |
| 1025 | 001269' | 254 08 0 0 | 002276* ILLEGAL WAITING TIME/3 |
| 1026 | 001270' | 051 33 0 0 | 003122* JRST QFRAME | ISEE SECTION D |
| 1027 | 001271' | 254 08 0 0 | 002276* NOTIME; | JSP Q,OUTCHR ;OUTPUT RETURN CHARACTER |
| 1028 | 001272' | 265 17 0 0 | 001240* JSP Q,INCHR | INPUT LINE FEED |
| 1029 | 001273' | 254 08 0 0 | 002276* JRST QFRAME | ISEE SECTION D |
| 1030 | 001274' | 265 17 0 0 | 001227* |

### SECTION 1. PROCESS EXTRA, ORDER, AND PHONIC STATEMENTS.

<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Microcode</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1035</td>
<td>001275'</td>
<td>201 02 0 0</td>
<td>002202* EXTRA: MOVEI B.2</td>
</tr>
<tr>
<td>1036</td>
<td>001276'</td>
<td>316 02 0 0</td>
<td>001217* CANN B.PART</td>
</tr>
<tr>
<td>1037</td>
<td>001277'</td>
<td>254 02 0 0</td>
<td>001326* JRST EXTOK</td>
</tr>
<tr>
<td>1038</td>
<td>001278'</td>
<td>265 17 0 0</td>
<td>001264* JSP Q,FRMLN2</td>
</tr>
<tr>
<td>1039</td>
<td>001279'</td>
<td>051 03 0 0</td>
<td>003197* TTCALL 3,[ASCII]</td>
</tr>
<tr>
<td>1040</td>
<td>001301'</td>
<td>251 03 0 0</td>
<td>003130* EXTRA STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/</td>
</tr>
<tr>
<td>1041</td>
<td>001302'</td>
<td>201 01 0 0</td>
<td>002011* EXTOK: MOVEI A.11</td>
</tr>
<tr>
<td>1042</td>
<td>001303'</td>
<td>254 02 0 0</td>
<td>001120* JRST QONOFF</td>
</tr>
<tr>
<td>1043</td>
<td>001304'</td>
<td>201 02 0 0</td>
<td>002002* ORDER: MOVEI B.2</td>
</tr>
<tr>
<td>1044</td>
<td>001305'</td>
<td>316 02 0 0</td>
<td>002276* CANN B.PART</td>
</tr>
<tr>
<td>1045</td>
<td>001306'</td>
<td>254 02 0 0</td>
<td>001111* JRST ORDOK</td>
</tr>
<tr>
<td>1046</td>
<td>001307'</td>
<td>265 17 0 0</td>
<td>001300* JSP Q,FRMLN2</td>
</tr>
<tr>
<td>1047</td>
<td>001308'</td>
<td>051 03 0 0</td>
<td>003202* TTCALL 3,[ASCII]</td>
</tr>
<tr>
<td>1048</td>
<td>001310'</td>
<td>251 03 0 0</td>
<td>003122* ORDER STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/</td>
</tr>
<tr>
<td>1049</td>
<td>001311'</td>
<td>201 01 0 0</td>
<td>002013* ORDOK: MOVEI A.13</td>
</tr>
<tr>
<td>1050</td>
<td>001312'</td>
<td>254 02 0 0</td>
<td>001120* JRST QONOFF</td>
</tr>
<tr>
<td>1051</td>
<td>001313'</td>
<td>201 02 0 0</td>
<td>002002* PHONIC: MOVEI B.2</td>
</tr>
<tr>
<td>1052</td>
<td>001314'</td>
<td>316 02 0 0</td>
<td>001355* CANN B.PART</td>
</tr>
<tr>
<td>1053</td>
<td>001315'</td>
<td>254 00 0 0</td>
<td>001320* JRST PHONOK</td>
</tr>
<tr>
<td>1054</td>
<td>001316'</td>
<td>265 17 0 0</td>
<td>001387* JSP Q,FRMLN2</td>
</tr>
<tr>
<td>1055</td>
<td>001317'</td>
<td>051 03 0 0</td>
<td>003135* TTCALL 3,[ASCII]</td>
</tr>
<tr>
<td>1056</td>
<td>001317'</td>
<td>251 03 0 0</td>
<td>003135* PHONIC STATEMENT DOES NOT FOLLOW ANSWER STATEMENT/</td>
</tr>
<tr>
<td>1057</td>
<td>001320'</td>
<td>474 04 0 0</td>
<td>000000* PHONOK: SETD D;</td>
</tr>
<tr>
<td>1058</td>
<td>001321'</td>
<td>201 01 0 0</td>
<td>002014* Movei A.14</td>
</tr>
<tr>
<td>1059</td>
<td>001322'</td>
<td>254 00 0 0</td>
<td>001121* JRST QONOFF</td>
</tr>
<tr>
<td>1060</td>
<td>001323'</td>
<td>265 17 0 0</td>
<td>001387* JSP Q,FRMLN2</td>
</tr>
</tbody>
</table>
ISECTION K.  PROCESS A MATCH STATEMENT.

1129

1130 00141$ 281 02 0 00 00022 MATCH: MOVE B,2

1131 00141$ 312 02 0 00 001324 CAKE B, PART: HAS ANSWER STATEMENT BEEN ENCOUNTERED?

1132 00141$ 234 00 0 00 001454 JST MDGR: NO, GO TYPE ERROR MESSAGE

1133 00141$ 262 00 0 00 001501 RECOV: MOVE B, A: SAVE CHARACTER IN AC-A

1134 00142$ 221 01 0 00 002023 JST * +2: PUT STATEMENT CODE IN AC-A

1135 00142$ 265 17 0 00 00140* MOVE A, B: PUT CHARACTER BACK IN AC-A

1136 00142$ 202 01 0 00 002022 JST * +2: INPUT CHARACTER

1137 00142$ 254 00 0 00 001425 JST * +2: INPUT CHARACTER

1138 00142$ 265 17 0 00 001401 JST * +2: INPUT CHARACTER

1139 00142$ 322 01 0 00 002040 CAKE A, 40

1140 00142$ 365 00 0 00 002011 CAKE A, 11

1141 00142$ 242 00 0 00 001424 JST * -3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER

1142 00143$ 265 17 0 00 001236 JST * X: INPUT MATCH NO.

1143 00143$ 391 02 0 00 001132 JST * X: IS NO, GREATER THAN 90?

1144 00143$ 254 00 0 00 001451 JST MERR: YES, GO TYPE ERROR MESSAGE

1145 00143$ 254 00 0 00 001452 JST * +2: INPUT CHARACTER

1146 00143$ 254 00 0 00 001424 JST * +2: INPUT CHARACTER

1147 00143$ 322 01 0 00 002040 CAKE A, 40

1148 00143$ 365 00 0 00 002011 CAKE A, 11

1149 00143$ 242 00 0 00 001424 JST * -3: IF CHARACTER IS A SPACE OR A TAB, GO INPUT NEXT CHARACTER

1150 00144$ 322 01 0 00 002041 CAKE A, 12: IF END OF STATEMENT?

1151 00144$ 254 00 0 00 001451 JST MERR: NO, GO TYPE ERROR MESSAGE

1152 00144$ 200 00 0 00 002020 JST MDGR: MOVE A, B: PUT MATCH NO. IN AD-A

1153 00144$ 271 31 2 00 001309 JST MDGR: ADD A, 40: ADD 40 TO IT

1154 00144$ 265 17 0 00 001421 JST MDGR: JSP G, OUTCHR: OUTPUT THE MATCH NO.

1155 00145$ 221 01 0 00 002015 MOVE A, 15

1156 00145$ 265 17 0 00 001444 JST MDGR: JSP G, OUTCHR: OUTPUT RETURN CHARACTER

1157 00145$ 269 17 0 00 001444 JST MDGR: JSP G, OUTCHR: INPUT LINE FEED

1158 00145$ 254 00 0 00 002076 JST MDGR: JST OFRAME: SEE SECTION D

1159 00145$ 265 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1160 00145$ 269 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1161 00145$ 254 00 0 00 001411 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1162 00145$ 265 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1163 00145$ 254 00 0 00 001411 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1164 00145$ 269 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1165 00145$ 254 00 0 00 001411 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1166 00145$ 269 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1167 00145$ 254 00 0 00 001411 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.

1168 00145$ 269 17 0 00 001406 JST MDGR: JSP G, FRMLIN: TYPE FRAME AND LINE NOS.
; SECTION I. PROCESS AN ACTION STATEMENT.

; ACTION: MOVE B, 3
; CAMM B, PART: IS THIS THE SECOND ACTION STATEMENT?
; JRST SGERR5: YES, GO TYPE ERROR MESSAGE
; MOVEM B, PART: SET PART NO. TO 3
; RECOV1: MOVE B, A: SAVE CHARACTER IN AC-A
; JSP Q, OUTCHR: OUTPUT <21>
; JSP Q, INCHR: INPUT CHARACTER BACK IN AC-A
; JRST #2: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
; CAIE A, 11: ENDF OF ACTION STATEMENT?
; JRST #11: NO, GO TYPE ERROR MESSAGE
; JRST #241: JSP Q, OUTCHR: OUTPUT RETURN CHARACTER
; JSP Q, INCHR: INPUT LINE FEED
; JRST QFRAME: SEE SECTION D
; SGERR1: JSP Q, FRML2: TYPE FRAME AND LINE NOS.
; TCALL #3, [ASCII] / MORE THAN ONE ACTION STATEMENT IN A SINGLE Q-FRAME/3
; JSP Q, FRML2: TYPE FRAME AND LINE NOS.
; TCALL #3, [ASCII] / ILLEGAL ACTION STATEMENT FORMAT/7
; JRST QFRAME: SEE SECTION D

; SECTION II. PROCESS ACTION ANSWER TAGS.

; NTAG: MOVE B, A: MOVE TAG TO AC-B
; JSP Q, INCHR: INPUT TAG DELIMITER
; MOVEI C, 2
; CAMLE C, PART: HAS AN ANSWER STATEMENT BEEN ENCOUNTERED?
; JSP Q, SYM: NO, GO TYPE ERROR MESSAGE (SECTION D)
; JRST ATAG1: IS THIS AN ANSWER TAG?
; JRST #12: NO, GO TO ATAG3
; TRO FLAGS=10: SET NUMERIC ANSWER FLAG
; JRST #2: SEE SECTION L
; ATAG1: MOVE C, 2
; CAMLE C, PART: HAS AN ANSWER STATEMENT BEEN ENCOUNTERED?
; JSP Q, SYM: NO, GO TYPE ERROR MESSAGE (SECTION D)
; JRST ATAG1: IS THIS AN ANSWER TAG?
; JRST #12: NO, GO TO ATAG3
; JRST #5: YES, GO TO TWO (SECTION L)
PROS MACRO 47(284)-.2 08/17 2-AUG-74 PAGE 1-26

1379 00:1761: 200 00 00 02031.
1380 00:1762: 201 01 00 02032.
1381 00:1763: 205 17 00 001732.
1382 00:1764: 220 01 00 000707.
1383 00:1765: 225 17 00 001733.
1384 00:1766: 228 01 00 000703.
1385 00:1767: 250 02 00 001721.
1386 00:1771: 269 17 00 001744.
1387 00:1771: 306 02 00 000794.
1388 00:1772: 254 22 00 001770.
1389 00:1773: 320 01 00 000811.
1390 00:1774: 305 01 00 000794.
1391 00:1775: 254 03 00 001770.
1392 00:1776: 326 01 00 000811.
1393 00:1777: 254 22 00 000802.
1394 00:1778: 254 17 00 001755.
1395 00:1779: 254 02 00 001772.
1396 00:1780: 254 17 00 000806.
1397 00:2023: 254 17 00 001770.
1398 00:2024: 254 00 00 002276.

SECTION 0. PROCESS AN L1 STATEMENT.

1404 0002033: 203 02 00 001757.
1405 0002034: 302 02 00 002303.
1406 0002035: 254 06 00 002034.
1407 0002036: 201 01 00 002013.
1408 0002037: 254 02 00 002016.

SECTION P. PROCESS FI AND DI STATEMENTS.

1414 0002042: 203 02 00 002035.
1415 0002043: 302 02 00 002033.
1416 0002044: 254 06 00 002034.
1417 0002045: 201 31 00 002035.
1418 0002046: 263 17 00 002022.
1419 0002047: 265 17 00 002023.
1420 0002048: 306 21 00 002015.
1421 0002049: 254 02 00 002011.
1422 0002050: 254 02 00 002011.
1423 0002051: 203 02 00 002001.
1424 0002052: 201 01 00 002001.
1425 0002053: 205 17 00 002016.
1426 0002054: 206 11 00 002022.
1427 0002055: 306 21 00 002015.
1428 0002056: 254 02 00 002011.
1429 0002057: 254 02 00 002011.
1430 0002058: 263 17 00 002022.
1431 0002059: 265 17 00 002023.
1432 0002060: 203 02 00 002011.
1433 0002061: 201 01 00 002011.
1434 0002062: 205 17 00 002016.
1435 0002063: 206 11 00 002022.
1436 0002064: 306 21 00 002015.
I SECTION T. PROCESS AN E1 STATEMENT.

1485 MOV B,PART  ;GET PART NO.
1486 TIMES 001H  ;HAS ACTION STATEMENT BEEN ENCOUNTERED?
1487 JST SQERR  ;IND. GO TYPE ERROR MESSAGE (SECTION P)
1488 TIMES 001H  ;MOV E1, A,175  ;PUT <175> IN AC-A
1489 TIMES 001H  ;JST FCMD  ;GO TO FCMD (SECTION P)

I SECTION U. PROCESS A G1 STATEMENT.

1490 MOV B,PART  ;GET PART NO.
1491 TIMES 001H  ;HAS ACTION STATEMENT BEEN ENCOUNTERED?
1492 JST SQERR  ;IND. GO TYPE ERROR MESSAGE (SECTION P)
1493 TIMES 001H  ;JSP Q, INCHR  ;INPUT CHARACTER
1494 TIMES 001H  ;JST A,15  ;END OF LINE?
1495 TIMES 001H  ;JST GRET  ;YES, GO TO GRET
1496 TIMES 001H  ;MOV E1, A,180  ;SAVE CHARACTER IN AC-A
1497 TIMES 001H  ;MOV E1, A,30  ;PUT A <30> IN AC-A
1498 TIMES 001H  ;JSP Q, OUTCHR  ;OUTPUT CHARACTER
1499 TIMES 001H  ;MOV E1, A,30  ;PUT CHARACTER BACK IN AC-A
1500 TIMES 001H  ;JST BCD1  ;GO TO BCD1 (SECTION V)
1501 TIMES 001H  ;GRET: JSP Q, INCHR  ;INPUT LINE FEED
1502 TIMES 001H  ;MOV E1, A,31  ;PUT <31> IN AC-A
1503 TIMES 001H  ;JSP Q, OUTCHR  ;OUTPUT CHARACTER
1504 TIMES 001H  ;MOV E1, A,15  ;PUT RETURN CHARACTER IN AC-A
1505 TIMES 001H  ;JST GFRAME  ;GO TO GFRAME (SECTION D)

I SECTION V. PROCESS A B1 STATEMENT.

1506 MOV B,PART  ;GET PART NO.
1507 TIMES 001H  ;HAS ACTION STATEMENT BEEN ENCOUNTERED?
1508 JST SQERR  ;IND. GO TYPE ERROR MESSAGE (SECTION P)
1509 TIMES 001H  ;MOV E1, A,7  ;PUT A <27> IN AC-A
1510 TIMES 001H  ;JSP Q, OUTCHR  ;OUTPUT CHARACTER
1511 TIMES 001H  ;MOV E1, A,11  ;INPUT CHARACTER
1512 TIMES 001H  ;JST 0  ;IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
1513 TIMES 001H  ;BCD11 CALL A,60  ;IF NOT A NUMERAL, GO TYPE ERROR MESSAGE
1514 TIMES 001H  ;JSP Q, INCHR  ;INPUT FRAME NO.
<table>
<thead>
<tr>
<th>PROC</th>
<th>MACRO 47(224)-7 00150 2-AUG-74 PAGE 1-30</th>
</tr>
</thead>
<tbody>
<tr>
<td>1601</td>
<td>002217T 265 17 0 00 02216T</td>
</tr>
<tr>
<td>1602</td>
<td>002220T 305 02 0 00 00144</td>
</tr>
<tr>
<td>1603</td>
<td>002221T 265 17 0 00 02217T</td>
</tr>
<tr>
<td>1604</td>
<td>002222T 305 02 0 00 00012</td>
</tr>
<tr>
<td>1605</td>
<td>002223T 265 17 0 00 02221T</td>
</tr>
<tr>
<td>1606</td>
<td>002224T 200 01 0 00 00022</td>
</tr>
<tr>
<td>1607</td>
<td>002225T 265 17 0 00 02216T</td>
</tr>
<tr>
<td>1608</td>
<td>002226T 201 01 0 00 00015</td>
</tr>
<tr>
<td>1609</td>
<td>002227T 265 17 0 00 02223T</td>
</tr>
<tr>
<td>1610</td>
<td>002230T 265 17 0 00 02216T</td>
</tr>
<tr>
<td>1611</td>
<td>002231T 305 01 0 00 00024</td>
</tr>
<tr>
<td>1612</td>
<td>002234T 254 30 0 00 02251</td>
</tr>
<tr>
<td>1613</td>
<td>002235T 305 01 0 00 00036</td>
</tr>
<tr>
<td>1614</td>
<td>002234T 254 20 0 00 000240</td>
</tr>
<tr>
<td>1615</td>
<td>002237T 265 17 0 00 02227T</td>
</tr>
<tr>
<td>1616</td>
<td>002238T 254 30 0 00 02230T</td>
</tr>
</tbody>
</table>

\section{Y. PROCESS A B-FRAME}

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1621</td>
</tr>
<tr>
<td>1622</td>
</tr>
<tr>
<td>1623</td>
</tr>
<tr>
<td>1624</td>
</tr>
<tr>
<td>1625</td>
</tr>
<tr>
<td>1626</td>
</tr>
<tr>
<td>1627</td>
</tr>
</tbody>
</table>

\section{Z. PROCESS A C-FRAME}

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1642</td>
</tr>
<tr>
<td>1643</td>
</tr>
<tr>
<td>1644</td>
</tr>
</tbody>
</table>

\section{AA. MAKE A SECOND PASS THRU THE LESSON TO COUNT THE NUMBER OF CHARACTERS PRECEDING EACH FRAME.}

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1662</td>
</tr>
<tr>
<td>1663</td>
</tr>
<tr>
<td>1664</td>
</tr>
<tr>
<td>1665</td>
</tr>
<tr>
<td>1666</td>
</tr>
<tr>
<td>1667</td>
</tr>
<tr>
<td>1668</td>
</tr>
<tr>
<td>1669</td>
</tr>
<tr>
<td>1670</td>
</tr>
<tr>
<td>1671</td>
</tr>
<tr>
<td>1672</td>
</tr>
<tr>
<td>Line</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>1803</td>
</tr>
<tr>
<td>1804</td>
</tr>
<tr>
<td>1805</td>
</tr>
<tr>
<td>1806</td>
</tr>
<tr>
<td>1807</td>
</tr>
<tr>
<td>1808</td>
</tr>
<tr>
<td>1809</td>
</tr>
<tr>
<td>1810</td>
</tr>
<tr>
<td>1811</td>
</tr>
<tr>
<td>1812</td>
</tr>
<tr>
<td>1813</td>
</tr>
<tr>
<td>1814</td>
</tr>
</tbody>
</table>

NO ERRORS DETECTED

PROGRAM BREAK IS 003325

5K CORE USED
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P2</td>
<td>1249#</td>
</tr>
<tr>
<td>P3</td>
<td>1284#</td>
</tr>
<tr>
<td>PAGE</td>
<td>515</td>
</tr>
<tr>
<td>PART</td>
<td>234#</td>
</tr>
<tr>
<td>PC1</td>
<td>236#</td>
</tr>
<tr>
<td>PC2</td>
<td>236#</td>
</tr>
<tr>
<td>PC3</td>
<td>236#</td>
</tr>
<tr>
<td>PCH</td>
<td>543</td>
</tr>
<tr>
<td>PGERR1</td>
<td>743</td>
</tr>
<tr>
<td>PGERR2</td>
<td>755</td>
</tr>
<tr>
<td>PHON</td>
<td>123</td>
</tr>
<tr>
<td>PHONIC</td>
<td>123</td>
</tr>
<tr>
<td>PHONOK</td>
<td>1053</td>
</tr>
<tr>
<td>POINT</td>
<td>249#</td>
</tr>
<tr>
<td>PRCS</td>
<td>251</td>
</tr>
<tr>
<td>PRCS2</td>
<td>251</td>
</tr>
<tr>
<td>PROC</td>
<td>371</td>
</tr>
<tr>
<td>PRO1</td>
<td>353</td>
</tr>
<tr>
<td>PRSFN</td>
<td>235#</td>
</tr>
<tr>
<td>PRSST</td>
<td>235#</td>
</tr>
</tbody>
</table>
TITLE CFRAIM

COMMENT 

WRITTEN BY GARRET A. VANDER LUIGT  COMPLETED 6-JUN-74

PART OF LPROS,SHR, VERSION 3

CFRAM,MAC PROCESSES C-FRAME STATEMENTS.  KEYWORDS IN EACH STATEMENT ARE REPLACED WITH SINGLE CHARACTER CODES.  ALGEBRAIC EXPRESSIONS ARE CONVERTED TO POLISH NOTATION, AND ADDRESSES ARE ASSIGNED TO VARIABLE NAMES.

CONTENTS OF CFRAIM,MAC

SECTION A.  CONTAINS ENTRY POINTS TO CFRAIM,MAC.

SECTION B.  PROCESSES LINE NUMBERS.

SECTION C.  PROCESSES ITERATION FIELDS.

SECTION D.  DETERMINES STATEMENT TYPE.

SECTION E.  PROCESSES RESERVE STATEMENTS.

SECTION F.  PROCESSES ARRAY STATEMENTS.

SECTION G.  PROCESSES INPUT STATEMENTS.

SECTION H.  PROCESSES TYPE STATEMENTS.

SECTION I.  PROCESSES ATYPE STATEMENTS.

SECTION J.  PROCESSES GOTO STATEMENTS.

SECTION K.  PROCESSES BI AND GI STATEMENTS.

SECTION L.  PROCESSES FILL STATEMENTS.

SECTION M.  PROCESSES IF STATEMENTS.

SECTION N.  PROCESSES ASSIGN STATEMENTS.

SECTION O.  POLISH NOTATION GENERATOR.

SECTION P.  PLACES VARIABLE IN SYMBOL TABLE.

SECTION Q.  PROCESSES SCALAR STATEMENTS.

SECTION R.  PROCESSES CALL STATEMENTS.
SECTION 5. PROCESSES RETURN STATEMENTS.

LIST OF CHARACTER CODES PRODUCED BY CFRAHM, 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;01&gt;</td>
<td>Precedes regular variable address</td>
</tr>
<tr>
<td>&lt;03&gt;</td>
<td>Input</td>
</tr>
<tr>
<td>&lt;05&gt;</td>
<td>ATYPE</td>
</tr>
<tr>
<td>&lt;07&gt;</td>
<td>B1</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;17&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;50&gt;</td>
<td>GI</td>
</tr>
<tr>
<td>&lt;60&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 CFRAHM, MAC

- CEXP - FROM PROS.MAC (SECTION L), LOCATED IN SECTION A
- CFRAHM - FROM PROS.MAC (SECTION Z), LOCATED IN SECTION A
- CSTATE - FROM PROS.MAC (SECTION W), LOCATED IN SECTION A
- TXTYP - FROM PROS.MAC (SECTION E), LOCATED IN SECTION A

EXTERNAL SUBROUTINES CALLED BY CFRAHM, MAC

- FLOUT - OUTPUTS THE FLOATING POINT NUMBER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION J)
- FRMLIN - TYPES FRAME AND LINE NUMBERS, LOCATED IN AUX4.MAC (SECTION F)
- II - INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4.MAC (SECTION H)
- II2 - INPUTS AN INTEGER FROM THE DISK INTO AC-B, LOCATED IN AUX4.MAC (SECTION H)
- INCHR - INPUTS A CHARACTER FROM THE DISK INTO AC-A, LOCATED IN AUX4.MAC (SECTION A)
- INFLO - INPUTS A FLOATING POINT NUMBER FROM THE DISK INTO AC-B, LOCATED IN AUX4.MAC (SECTION I)
100 - OUTPUTS THE INTEGER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION C)
1008 - OUTPUTS THE OCTAL NO. IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION C)
1017 - OUTPUTS THE INTEGER IN AC-A TO A USER'S TERMINAL, LOCATED IN AUX4.MAC (SECTION E)
1018 - OUTPUTS THE CHARACTER IN AC-A TO THE DISK, LOCATED IN AUX4.MAC (SECTION B)

EXTERNAL MEMORY REFERENCES (ALL LOCATED IN PRSLOW.MAC)
ADDR = ADDRESS REGISTER USED TO ASSIGN MEMORY LOCATIONS TO VARIABLES APPEARING IN C-FRAME STATEMENTS
ASGN = AND SIGN COUNT (10 WORDS)
ERRTOT = TOTAL NO. OF ERRORS DETECTED
FRMNAM = CURRENT FRAME NO.
GOTOCH = GOTO NUMBER BITS
HOLD = USED TO HOLD OPERATOR PRECEDING AN ARRAY
ITREV = NO. OF ITERATION VARIABLES IN ITERATION FIELD
LEVEL = PARENTHESIS LEVEL
LINE = CURRENT LINE NO.
LINENO = LINE NUMBER BITS
OSTACK = OPERATOR STACK (32 WORDS)
PC3 = SUBROUTINES STORE RETURN ADDRESS HERE
PC4 = SUBROUTINES STORE RETURN ADDRESS HERE
PC5 = SUBROUTINES STORE RETURN ADDRESS HERE
PSTACK = POLISH STACK (72 WORDS)
PTYPE = POLISH TYPE STACK (72 WORDS)
RSVNUM = TOTAL NO. OF WORDS RESERVED FOR STUDENT CALC MODE VARIABLE STORAGE
SSTACK = STRING STACK (72 WORDS)
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVEM OPC4</td>
<td>SAVE RETURN ADDRESS</td>
</tr>
<tr>
<td>TRF FLAGS,120</td>
<td>SET RELATIONAL SYMBOL AND O-FRAME FLAGS</td>
</tr>
<tr>
<td>MOVEL G,1</td>
<td>SET STACK INDEX TO 1</td>
</tr>
<tr>
<td>JSF Q,POLY3</td>
<td>CONVERT EXPRESSION TO POLISH FORM (SECT. O)</td>
</tr>
<tr>
<td>CAIL A,15</td>
<td></td>
</tr>
<tr>
<td>JST @PC4</td>
<td>IF CHARACTER IS &lt;RETURN&gt; OR &quot;,&quot;, RETURN</td>
</tr>
</tbody>
</table>
SECTION B. PROCESSES LINE NUMBERS.

<table>
<thead>
<tr>
<th>Line 213</th>
<th>083615</th>
<th>265</th>
<th>17</th>
<th>00</th>
<th>000300</th>
<th>JSP 0, FRM-LIN</th>
<th>ELSE: TYPE FRAME AND LINE NOS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line 214</td>
<td>083616</td>
<td>265</td>
<td>17</td>
<td>00</td>
<td>000300</td>
<td>TDCALL J, LASCIE /</td>
<td></td>
</tr>
<tr>
<td>Line 215</td>
<td>083617</td>
<td>25</td>
<td>03</td>
<td>00</td>
<td>00021</td>
<td>ANSWER EXPRESSION NOT DELIMITED BY A COMMA</td>
<td></td>
</tr>
<tr>
<td>Line 216</td>
<td>083618</td>
<td>254</td>
<td>02</td>
<td>00</td>
<td>00021</td>
<td>JRT 9, PC4</td>
<td>RETURN TO PROG, MAC</td>
</tr>
<tr>
<td>Line 217</td>
<td>083619</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00021</td>
<td>CSTATE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 218</td>
<td>083620</td>
<td>660</td>
<td>20</td>
<td>00</td>
<td>00022</td>
<td>TRD FLAGS, 20</td>
<td>SET 0-FRAME FLAG</td>
</tr>
<tr>
<td>Line 219</td>
<td>083621</td>
<td>254</td>
<td>02</td>
<td>00</td>
<td>00021</td>
<td>JRT 3, IC4</td>
<td>GO TO ITER (SECTION C)</td>
</tr>
<tr>
<td>Line 220</td>
<td>083622</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 221</td>
<td>083623</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 222</td>
<td>083624</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 223</td>
<td>083625</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 224</td>
<td>083626</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 225</td>
<td>083627</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 226</td>
<td>083628</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 227</td>
<td>083629</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 228</td>
<td>083630</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 229</td>
<td>083631</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 230</td>
<td>083632</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 231</td>
<td>083633</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 232</td>
<td>083634</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 233</td>
<td>083635</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 234</td>
<td>083636</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 235</td>
<td>083637</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 236</td>
<td>083638</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 237</td>
<td>083639</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 238</td>
<td>083640</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 239</td>
<td>083641</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 240</td>
<td>083642</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 241</td>
<td>083643</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 242</td>
<td>083644</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 243</td>
<td>083645</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 244</td>
<td>083646</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 245</td>
<td>083647</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 246</td>
<td>083648</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 247</td>
<td>083649</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 248</td>
<td>083650</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 249</td>
<td>083651</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 250</td>
<td>083652</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 251</td>
<td>083653</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 252</td>
<td>083654</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 253</td>
<td>083655</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 254</td>
<td>083656</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 255</td>
<td>083657</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 256</td>
<td>083658</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 257</td>
<td>083659</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 258</td>
<td>083660</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 259</td>
<td>083661</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 260</td>
<td>083662</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 261</td>
<td>083663</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 262</td>
<td>083664</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 263</td>
<td>083665</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 264</td>
<td>083666</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
<tr>
<td>Line 265</td>
<td>083667</td>
<td>222</td>
<td>17</td>
<td>00</td>
<td>00022</td>
<td>TXTYPE</td>
<td>MOVER 0, PC4</td>
</tr>
</tbody>
</table>
SECTION C. PROCESSES ITERATION FIELDS.

ITER: JSP 0,INCHR | INPUT CHARACTER

ITER2: CAI E,11 | IF CHARACTER IS A SPACE OR TAB, GO_INPUT NEXT CHARACTER

ITER3: CAI E, "C" | IS CHARACTER A "C"?

ITER4: JSP 0,INCHR | INPUT CHARACTER

ITER5: CAI E,11 | IF CHARACTER IS A SPACE OR TAB, GO_INPUT NEXT CHARACTER

ITER6: SETM 17TLREV | SET ITERATION LEVEL TO ZERO

ITER7: CAIN A,"S" | END OF ITERATION FIELD

ITER8: JSP 0,INCHR | INPUT CHARACTER

ITER9: CAI E,11 | IF CHARACTER IS A SPACE OR TAB, GO_INPUT NEXT CHARACTER

ITER10: SETM 17TLREV | SET ITERATION LEVEL TO ZERO

ITER11: CAIN A,"S" | END OF ITERATION FIELD

ITER12: JSP 0,INCHR | INPUT CHARACTER

ITER13: CAI E,11 | IF CHARACTER IS A SPACE OR TAB, GO_INPUT NEXT CHARACTER

ITER14: SETM 17TLREV | SET ITERATION LEVEL TO ZERO

ITER15: CAIN A,"S" | END OF ITERATION FIELD

ITER16: JSP 0,INCHR | INPUT CHARACTER

ITER17: CAI E,11 | IF CHARACTER IS A SPACE OR TAB, GO_INPUT NEXT CHARACTER

ITER18: SETM 17TLREV | SET ITERATION LEVEL TO ZERO

ITER19: CAIN A,"S" | END OF ITERATION FIELD
ISECTION E. PROCESSES RESERVE STATEMENTS,

032330'  602  26  00  00  003220  RSRV1: TRUE FLAGS  20 JIS THIS A Q-FRAME?
032331'  254  26  00  00  003254  JRS ASERVER2: YES, GO TYPE ERROR MESSAGE
032332'  254  17  00  00  003260  JSP Q, INCHR  INPUT CHARACTER
032333'  302  21  00  00  000011  CAI A 11
032334'  306  21  00  00  000040  CAI A 40
032335'  254  20  00  00  003322  JSP Q, INCHR  INPUT NUMBER OF WORDS RESERVED FOR STUDENTS
032336'  254  17  00  00  003361  JSP Q, 112  IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
032337'  254  17  00  00  003411  JRS T 2
032338'  254  17  00  00  003324  JSP Q, INCHR  INPUT CHARACTER
032339'  254  21  00  00  000011  CAI A 11
032340'  254  21  00  00  000040  CAI A 40
032341'  254  21  00  00  000040  CAI A 40
032342'  254  21  00  00  000040  CAI A 40
032343'  254  21  00  00  003401  JRS T 3
032344'  254  21  00  00  000015  CAI A 15: END OF LINE
032345'  254  21  00  00  003511  JRS ASERVER: YES, GO TYPE ERROR MESSAGE
032346'  254  21  00  00  003511  JSP Q, FRMLIN: ITYPE FRAME AND LINE NOS.
032347'  254  17  00  00  003574  JSP Q, OUTCHR: OUTPUT <F>
032348'  254  17  00  00  003574  JRS LINNUM: IGO TO LINNUM (SECTION B)
032349'  254  17  00  00  003574  JRS LINNUM: IGO TO LINNUM (SECTION B)
032350'  254  17  00  00  003574  JRS LINNUM: IGO TO LINNUM (SECTION B)

ISECTION F. PROCESSES ARRAY STATEMENTS,

032351'  682  20  00  00  003220  ARRAY T BLE FLAGS  20 JIS THIS A Q-FRAME?
032352'  254  00  00  0004761  JRS ASERVER2: YES, GO TYPE ERROR MESSAGE
<table>
<thead>
<tr>
<th>Line No.</th>
<th>OP Code</th>
<th>Instruction Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>470</td>
<td>222341</td>
<td>JSP Q.INCHR  IELSE, INPUT CHARACTER</td>
</tr>
<tr>
<td>471</td>
<td>222342</td>
<td>Q.A11</td>
</tr>
<tr>
<td>472</td>
<td>222343</td>
<td>Q.A12</td>
</tr>
<tr>
<td>473</td>
<td>222344</td>
<td>JIRST -3</td>
</tr>
<tr>
<td>474</td>
<td>222345</td>
<td>JIRST -4</td>
</tr>
<tr>
<td>475</td>
<td>222346</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>476</td>
<td>222347</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>477</td>
<td>222348</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>478</td>
<td>222349</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>479</td>
<td>222350</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>480</td>
<td>222351</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>481</td>
<td>222352</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>482</td>
<td>222353</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>483</td>
<td>222354</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>484</td>
<td>222355</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>485</td>
<td>222356</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>486</td>
<td>222357</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>487</td>
<td>222358</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>488</td>
<td>222359</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>489</td>
<td>222360</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>490</td>
<td>222361</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>491</td>
<td>222362</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>492</td>
<td>222363</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>493</td>
<td>222364</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>494</td>
<td>222365</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>495</td>
<td>222366</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>496</td>
<td>222367</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>497</td>
<td>222368</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>498</td>
<td>222369</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>499</td>
<td>222370</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>500</td>
<td>222371</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>501</td>
<td>222372</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>502</td>
<td>222373</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>503</td>
<td>222374</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>504</td>
<td>222375</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>505</td>
<td>222376</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>506</td>
<td>222377</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>507</td>
<td>222378</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>508</td>
<td>222379</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>509</td>
<td>222380</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>510</td>
<td>222381</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>511</td>
<td>222382</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>512</td>
<td>222383</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>513</td>
<td>222384</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>514</td>
<td>222385</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>515</td>
<td>222386</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>516</td>
<td>222387</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>517</td>
<td>222388</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>518</td>
<td>222389</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>519</td>
<td>222390</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>520</td>
<td>222391</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>521</td>
<td>222392</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>522</td>
<td>222393</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>523</td>
<td>222394</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>524</td>
<td>222395</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>525</td>
<td>222396</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>526</td>
<td>222397</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>527</td>
<td>222398</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>528</td>
<td>222399</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>529</td>
<td>222400</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>530</td>
<td>222401</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>531</td>
<td>222402</td>
<td>JIRST AERR</td>
</tr>
<tr>
<td>532</td>
<td>222403</td>
<td>JIRST AERR</td>
</tr>
</tbody>
</table>
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
CRA \ "A": IF CHARACTER IS AN AMD?
JST AFHERR: YES, GO TYPE ERROR MESSAGE
JSP G.112: INPUT 1ST FORMAT NO.
CAIE A, "$": IS NO. TERMINATED BY A "$", "?"
JST RERM: NO, GO TYPE ERROR MESSAGE (SECTION H)
MOVE A,B: INPUT NO. IN AC-A
JSP G.10D: OUTPUT FORMAT NO.
MOVEI A, "": OUTPUT A, ""
JSP G.OUTCHR: OUTPUT 2ND FORMAT NO.
JST + 2: INPUT CHARACTER
CAIE A, A: INPUT CHARACTER
CAIE A, 48: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
CAIE A, A: IF CHARACTER IS A CHARACTER "A"?
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
CAIE A, 48: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
CAIE A, A: END OF LINE?
JST AFHERR: YES, GO TO AFHERR
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
AFHERR: JSP G.FRMLIN: TYPE FRAME AND LINE NO.
JST + 2: CALL 3, (ASCII / "A" FORMAT MAY NOT BE USED IN TYPE STATEMENT)
JST LINVUM: GO TO LINVUM (SECTION B)

SECTION J: PROCESSES GOTO STATEMENTS.

GOTO: MOVEI A, 10: INPUT GOTO STATEMENT CODE IN AC-A
JSP G. OUTCHER: OUTPUT CODE - (10)
JSP G. INCHR: INPUT CHARACTER
CAIE A, 48: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
JSP G.112: INPUT LINE NO.
JST + 2: IF NO. IS ZERO, GO TYPE ERROR MESSAGE (SECTION B)
JST RERM: NO, GO TYPE ERROR MESSAGE (SECTION B)
JST + 2: INPUT CHARACTER
CAIE A, A: INPUT CHARACTER
JST + 3: IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
I!§w

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


I.

SECTION 0. POLISH NOTATION GENERATOR.

0  MOVEM  G,PC5  ;SAVE RETURN ADDRESS

JST  RVAR1  ;GO TO RVAR1

0  MOVEM  G,PC5  ;SAVE RETURN ADDRESS

JST  AVAR  ;GO TO AVAR

0  MOVEM  G,PC5  ;SAVE RETURN ADDRESS

JST  A VAR  ;GO TO AVAR

0  MOVEM  G,PC5  ;SAVE RETURN ADDRESS

JST  POLY1  ;GO TO POLY

;IS THIS THE ERROR VARIABLE?

JST  ,",+  ;YES, JUMP

CAIE  C,","  ;IS THIS THE DEVICE VARIABLE?

JST  SVERR  ;NO, GO TYPE ERROR MESSAGE

JSP  G, POLLY5  ;CONVERT STATEMENT TO POLISH FORM

JST  +  ;END OF LINE?

JSP  G,DUTCHR  ;OUTPUT RETURN

JST  TINCHR  ;INPUT LINE FEED

JST  LINNUM  ;GO TO LINNUM (SECTION B)

JSP  G,FRMLIN  ;TYPE FRAME AND LINE NOS.

TTCALL  3,CASIZ/  

A VALUE MAY NOT BE ASSIGNED TO THAT SPECIAL VARIABLE/3

MOVEM  G,PC5  ;SAVE RETURN ADDRESS

MOVI  G,55  ;PUT VARIABLE CODE IN AC-B

MOVEM  G,PC5  ;SAVE RETURN ADDRESS

MOVI  G,55  ;PUT VARIABLE CODE IN AC-B

MOVEI  B,"V"  ;PUT "V" IN STRING TYPE STACK

AQJG, POL2  ;GO TO POL2

MOVEM  G,PC5  ;SAVE RETURN ADDRESS

TRE FLACS,55B  ;CLEAR RELATIONAL SYMBOL AND INDEX FLAGS

MOVEI  A,"B"  ;PUT "B" IN STRING TYPE STACK

MOVEI  G,1  ;INPUT CHARACTER

JSP  G, INCHR  

POLLY1:  ;INPUT CHARACTER

JST  ,",+  ;YES, GO TO POLLY1

CAIE  C,","  ;IS CHARACTER A UNARY PLUS?

JST  LPAR  ;YES, GO TO LPAR

CAIN  A,"+"  ;IS CHARACTER A "+"?

CAIN  A,"-"  ;IS CHARACTER A "-"?

JST  LPAR  ;YES, GO TO LPAR

JST  CONST  ;YES, GO TO CONST

JST  UMINUS  ;YES, GO TO UMINUS

MOVEI  G,1  ;INPUT CHARACTER

JSP  G, INCHR  

POLLY1:  ;INPUT CHARACTER

JST  ,",+  ;YES, GO TO POLLY1

CAIE  C,","  ;IS CHARACTER A UNARY PLUS?

JST  LPAR  ;YES, GO TO LPAR

CAIN  A,"+"  ;IS CHARACTER A "+"?

CAIN  A,"-"  ;IS CHARACTER A "-"?

JST  LPAR  ;YES, GO TO LPAR

JST  CONST  ;YES, GO TO CONST

JST  UMINUS  ;YES, GO TO UMINUS

MOVEI  G,1  ;INPUT CHARACTER

JSP  G, INCHR  

POLLY1:  ;INPUT CHARACTER

JST  ,",+  ;YES, GO TO POLLY1

CAIE  C,","  ;IS CHARACTER A UNARY PLUS?

JST  LPAR  ;YES, GO TO LPAR

CAIN  A,"+"  ;IS CHARACTER A "+"?

CAIN  A,"-"  ;IS CHARACTER A "-"?

JST  LPAR  ;YES, GO TO LPAR

JST  CONST  ;YES, GO TO CONST

JST  UMINUS  ;YES, GO TO UMINUS

MOVEI  G,1  ;INPUT CHARACTER

JSP  G, INCHR  

POLLY1:  ;INPUT CHARACTER

JST  ,",+  ;YES, GO TO POLLY1

CAIE  C,","  ;IS CHARACTER A UNARY PLUS?

JST  LPAR  ;YES, GO TO LPAR

CAIN  A,"+"  ;IS CHARACTER A "+"?

CAIN  A,"-"  ;IS CHARACTER A "-"?

JST  LPAR  ;YES, GO TO LPAR

JST  CONST  ;YES, GO TO CONST

JST  UMINUS  ;YES, GO TO UMINUS

MOVEI  G,1  ;INPUT CHARACTER

JSP  G, INCHR  

POLLY1:  ;INPUT CHARACTER

JST  ,",+  ;YES, GO TO POLLY1

CAIE  C,","  ;IS CHARACTER A UNARY PLUS?
ILLEGAL

JST LINNUM
1GO TO LINNUM (SECTION B)

SPCVAR

SET B

JST Q, INCHR
1INPUT CHARACTER

CAIN A, "R"
1IS CHARACTER AN "R"?

MNVN B,1
1YES, SET VARIABLE CODE TO -1

CAIN A, "A"
1IS CHARACTER AN "A"?

MNVN B,2
1YES, SET VARIABLE CODE TO -2

CAIN A, "T"
1IS CHARACTER A "T"?

MNVN B,3
1YES, SET VARIABLE CODE TO -3

SPCVAR

SET Z

JST Q, INCHR
1INPUT CHARACTER

CAIN A, "R"
1IS CHARACTER AN "R"?

MNVN B,1
1YES, SET VARIABLE CODE TO -1

CAIN A, "A"
1IS CHARACTER AN "A"?

MNVN B,2
1YES, SET VARIABLE CODE TO -2

CAIN A, "T"
1IS CHARACTER A "T"?

MNVN B,3
1YES, SET VARIABLE CODE TO -3

SPCVAR

SET Z

JST Q, INCHR
1INPUT CHARACTER

CAIN A, "R"
1IS CHARACTER AN "R"?

MNVN B,1
1YES, SET VARIABLE CODE TO -1

CAIN A, "A"
1IS CHARACTER AN "A"?

MNVN B,2
1YES, SET VARIABLE CODE TO -2

CAIN A, "T"
1IS CHARACTER A "T"?

MNVN B,3
1YES, SET VARIABLE CODE TO -3
1538 022334 322 31 00 020275 CAIE A, "e"
1539 022335 326 31 00 020274 CAIE A, "e"
1540 022336 254 32 00 020295 JST RELOP 1 IF CHARACTER IS "=" OR "<", GO TO RELOP
1541 022337 326 31 00 020276 CAIN A, "=
1542 022338 254 32 00 020295 JST RELOP 1 IF CHARACTER IS "=" OR "<", GO TO RELOP
1543 022339 326 32 00 020291 JST ENDOP 1 ELSE, GO TO ENDOP
1544 022340 202 31 00 0702239 OPERI MOVEM A,STACK (G) INPUT OPERATOR IN STRING STACK
1545 022341 202 32 00 020290 MOVE B,A 1 SAVE THE CHARACTER IN AC-A
1546 022342 202 31 00 0702117 MOVEI A, "O" 1 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
1547 022343 202 31 00 0702236 MOVEM A, TYPE (G) 1 INPUT "O" IN STRING TYPE STACK
1548 022344 269 17 00 0702137 JSP G,INCHR 1 INPUT NEXT CHARACTER
1549 022345 322 31 00 020211 CAIE A, "e"
1550 022346 326 31 00 020240 CAIN A, "e"
1551 022347 254 32 00 020295 JST RELOP 1 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
1552 022348 326 32 00 020291 JST ENDOP 1 ELSE, GO TO ENDOP
1553 022349 344 27 00 0202277 ADJA G,POL2 1 IF CHARACTER WAS "J" OR ")", GO TO POL2
1554 022350 344 27 00 0702171 ADJA G,POL1 1 ELSE, GO TO POL1
1555 022351 622 32 00 0202260 COMMA TANE FLAGS,200 1 IS INDEX FLAG SET?
1556 022352 262 32 00 0202321 JST OPER 1 YES, GO TO OPER
1557 022353 201 03 00 0202351 ENDP1 MOVEM C,TYPE (G) 1 PUT "E" IN STRING TYPE STACK
1558 022354 202 20 00 0202235 JST POL3 1 GO TO POL3
1559 022355 629 72 00 0202729 RPARI TANE FLAGS,200 1 IS INDEX FLAG SET?
1560 022356 284 02 00 020295 JST OPER 1 NO, GO TO OPER
1561 022357 326 23 00 020293 TRE FLAGS,230 1 CLEAR INDEX FLAG
1562 022358 221 21 00 0202135 MOVEI A, "J" 1 REPLACE "J" BY "J"
1563 022359 254 22 00 0202321 JST OPER 1 GO TO OPER
1564 022360 202 00 00 0202260 RELOPI TANE FLAGS,130 1 IS RELATIONAL SYMBOL LEGAL?
1565 022361 284 23 00 0202311 JST OPER 1 NO, GO TYPE ERROR MESSAGE
1566 022362 622 32 00 0202310 TANE FLAGS,130 1 SET RELATIONAL SYMBOL LEGAL FLAG
1567 022363 284 20 00 0202310 TANE FLAGS,130 1 SET RELATIONAL SYMBOL LEGAL FLAG
1568 022364 201 21 00 0202137 JST OPER 1 NO, GO TO OPER
1569 022365 202 00 00 0202260 RELERRI JSP G,FRMLN 1 TYPE FRAME AND LINE NOS.
1570 022366 202 00 00 0203171 TICALL 3,ICASC1 / MORE THAN ONE RELATIONAL SYMBOL IN A SINGLE EXPRESSION/
1571 022367 202 00 00 0202361 JST LINNUM 1 GO TO LINNUM (SECTION B)
1572 022368 202 00 00 0203171 ERCERRI JSP G,FRMLN 1 TYPE FRAME AND LINE NOS.
1573 022369 202 00 00 0203614 TICALL 3,ICASC1 / ILLEGAL RELATIONAL SYMBOL/
<table>
<thead>
<tr>
<th>CFRAH</th>
<th>MACRO</th>
<th>47(284)-7 02193 2-AUG-74 PAGE 1-34</th>
</tr>
</thead>
<tbody>
<tr>
<td>1803</td>
<td>002770'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT 1ST DIGIT OF ADDRESS (BASE 80)</td>
</tr>
<tr>
<td>1804</td>
<td>002771'</td>
<td>MOVE A, B ( ; ) ( JSP. Q, OUTCHR ) ( ; ) OUTPUT 2ND DIGIT OF ADDRESS (BASE 80)</td>
</tr>
<tr>
<td>1805</td>
<td>002772'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) GO TO OUTI</td>
</tr>
<tr>
<td>1806</td>
<td>002773</td>
<td>( SCOUTY ) ( MOV.E1 ) ( A,20 ) ( ; ) INPUT SPECIAL VARIABLE CODE ( &lt;20&gt; ) IN AC-A</td>
</tr>
<tr>
<td>1807</td>
<td>002774</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT &lt;22&gt;</td>
</tr>
<tr>
<td>1808</td>
<td>002775'</td>
<td>MOV.EH, P, STACK ( (F) ) ( IGET ) ABSOLUTE VALUE OF VARIABLE CODE</td>
</tr>
<tr>
<td>1809</td>
<td>002776'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT CODE FOR THIS VARIABLE</td>
</tr>
<tr>
<td>1810</td>
<td>002777'</td>
<td>( SP.COUTY ) ( MOVE1 ) ( A,21 ) ( ; ) INPUT ARRAY VARIABLE CODE ( &lt;21&gt; ) IN AC-A</td>
</tr>
<tr>
<td>1811</td>
<td>002778'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT CODE</td>
</tr>
<tr>
<td>1812</td>
<td>002779'</td>
<td>( HR.R2 ) ( A, P, STACK ( (F) ) ( IGET ) ADDRESS OF VARIABLE FROM POLISH STACK</td>
</tr>
<tr>
<td>1813</td>
<td>00277A'</td>
<td>I( D I V I A ) ( A,1090 )</td>
</tr>
<tr>
<td>1814</td>
<td>00277B'</td>
<td>ADD.I A ( A,40 )</td>
</tr>
<tr>
<td>1815</td>
<td>00277C'</td>
<td>ADD.I B ( A,40 )</td>
</tr>
<tr>
<td>1816</td>
<td>00277D'</td>
<td>ADD.I A, B ( A,40 )</td>
</tr>
<tr>
<td>1817</td>
<td>00277E'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT 1ST DIGIT OF ADDRESS (BASE 80)</td>
</tr>
<tr>
<td>1818</td>
<td>00277F'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT 2ND DIGIT OF ADDRESS (BASE 80)</td>
</tr>
<tr>
<td>1819</td>
<td>002780'</td>
<td>( ML.RZ ) ( A, P, STACK ( (F) ) ( IGET ) NO, OF COLUMNS IN THIS ARRAY</td>
</tr>
<tr>
<td>1820</td>
<td>002781'</td>
<td>IDIVI A ( A,1090 )</td>
</tr>
<tr>
<td>1821</td>
<td>002782'</td>
<td>ADD.I A ( A,40 )</td>
</tr>
<tr>
<td>1822</td>
<td>002783'</td>
<td>ADD.I B ( A,40 )</td>
</tr>
<tr>
<td>1823</td>
<td>002784'</td>
<td>MOV.EB A ( A,40 )</td>
</tr>
<tr>
<td>1824</td>
<td>002785'</td>
<td>MOV.EH, P, STACK ( (F) ) ( IGET ) ADDRESS OF COLUMN COUNT (BASE 80)</td>
</tr>
<tr>
<td>1825</td>
<td>002786'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT 1ST DIGIT OF COLUMN COUNT (BASE 80)</td>
</tr>
<tr>
<td>1826</td>
<td>002787'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT 2ND DIGIT OF COLUMN COUNT (BASE 80)</td>
</tr>
<tr>
<td>1827</td>
<td>002788'</td>
<td>( JSP. Q, OUTCHR ) ( ; ) OUTPUT &lt;22&gt; IN AC-A</td>
</tr>
<tr>
<td>1828</td>
<td>002789'</td>
<td>( SP.COUTY ) ( MOVE1 ) ( A,21 ) ( ; ) INPUT CHARACTER THAT TERMINATED EXPRESSION IN AC-A</td>
</tr>
<tr>
<td>1829</td>
<td>00278A'</td>
<td>I( D I V I A ) ( A,9PC5 ) ( ; ) RETURN</td>
</tr>
</tbody>
</table>

!SECTION P. PLACE VARIABLE IN SYMBOL TABLE AND OUTPUT ITS ASSIGNED ADDRESS.

| 1830  | 00278B' | \( ASVARI \) MOV.EM D, PC3 \( ; \) SAVE RETURN ADDRESS |
| 1831  | 00278C' | MOV.EA, A \( ; \) SAVE THE CHARACTER IN AC-A |
| 1832  | 00278D' | MOV.EB, SYMIDX \( ; \) GET POINTER TO END OF SYMBOL TABLE |
| 1833  | 00278E' | \( AS.V1 \) \( ML.RZ \) \( A, SYMTAB \( (D) \) \( IGET \) SYMBOL FROM SYMBOL TABLE |
| 1834  | 00278F' | \( GA.MN \) A, G \( ; \) IGET VARIABLE NAME SAME AS SYMBOL? |
| 1835  | 002790' | J\( RST \) \( A,43 \) \( ; \) YES, JUMP |
| 1836  | 002791' | J\( RST \) \( A,40 \) \( ; \) NO, CHECK NEXT SYMBOL, IF ANY |
| 1837  | 002792' | \( AS.V2 \) \( HR.R2 \) \( A, SYMTAB \( (D) \) \( IGET \) ADDRESS OF VARIABLE |
| 1838  | 002793' | IDDIVI A \( A,9PC5 \) \( ; \) RETURN |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTION R. PROCESSES CALL STATEMENTS.

CALL1: MOVE I, "C"  
JSP G, D, OUTCHR  
OUTPUT CALL STATEMENT CODE - "C"

CALL2: MOVE I, "L"  
JSP G, D, INCHR  
INPUT CHARACTER

CALL3: MOVE I, "R"  
JSP G, D, INCHR  
INPUT NEXT CHARACTER OF LESSON NAME

CALL4: MOVE I, "S"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER

CALL5: MOVE I, "T"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A

CALL6: MOVE I, "U"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A

CALL7: MOVE I, "V"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A

CALL8: MOVE I, "W"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A

CALL9: MOVE I, "X"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A

CALL10: MOVE I, "Y"  
JSP G, D, OUTCHR  
OUTPUT CHARACTER IN AC-A
1962 003133 302 01 0 0 002071 CAIE A.11
1963 003134 306 00 0 0 002074 CAIE A.40
1964 003133 254 30 0 0 223132 JRST +3 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
1965 003136 403 02 0 0 220220 SETE B,
1966 003137 254 03 0 0 003141 JRST +2
1967 003142 265 17 0 0 003142 JSP Q, INCHR INPUT CHARACTER
1968 003141 320 31 0 0 002748 CAIE A."R"
1969 003142 320 31 0 0 002767 CAIE A."M"
1970 003143 254 02 0 0 003147 JRST ENDPN IF NOT AN OCTAL DIGIT, GO TO ENDPN
1971 003144 221 32 0 0 002729 IMULI B,10 RELOAD DIGIT TO NO. IN AC-B LEFTH DIGIT
1972 003145 221 71 0 0 007729 100 DIGIT TO NO. IN AC-B
1973 003146 254 03 0 0 003147 JRST INPEN INPUT CHARACTER
1974 003147 207 10 0 0 003121 ENDPN MOVE H,A SAVE THE CHARACTER IN AC-A
1975 003150 220 31 0 0 003122 JSP Q, JDOS OUTPUT THE OCTAL NO.
1976 003151 265 17 0 0 003140 MOVE A,H INPUT CHARACTER BACK IN AC-A
1977 003152 220 01 0 0 002710 JSP Q, INCHR INPUT CHARACTER
1978 003153 254 73 0 0 003155 JRST +2
1979 003154 265 17 0 0 003148 CAIE A.11
1980 003155 322 01 0 0 002711 CAIE A.40
1981 003156 325 31 0 0 003154 JRST +2 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
1982 003157 254 00 0 0 003154 CAIE A,"M" IS CHARACTER A COMMA?
1983 003158 326 01 0 0 003154 JRST +2 CAIE A,"M" IS CHARACTER A "M"?
1984 003159 344 05 0 0 003131 AOJA E, PR PPRG+1 YES, GO INPUT NEXT NUMBER
1985 003160 322 01 2 0 003135 CAIE A."M" IS CHARACTER A "M"?
1986 003161 254 00 0 0 0031727 JRST FENRR2 NO, GO TYPE ERROR MESSAGE
1987 003162 308 05 0 0 003172 CAIE E,"M" MORE THAN 2 NUMBERS?
1988 003163 254 00 0 0 003172 JRST FENRR2 YES, GO TYPE ERROR MESSAGE
1989 003164 265 17 0 0 003154 JRST +2 INPUT CHARACTER
1990 003165 265 17 0 0 003154 JRST +2 INPUT CHARACTER
1991 003166 265 17 0 0 003154 JRST +2 INPUT CHARACTER
1992 003167 302 21 2 0 003140 CAIE A.15 END OF LINE?
1993 003168 254 22 2 0 003175 JRST +2 NO, GO INPUT NEXT CHARACTER
1994 003169 265 17 0 0 003125 FENRR2: JSP Q, FRMLIN TYPE FRAME AND LINE NOS.
1995 003170 251 03 2 20 003142 PROJ-PRG NUMBERS NOT IN PROPER FORMAT?
1996 003171 254 20 0 0 003166 JRST LINNUM GO TO LINNUM (SECTION B)
1997 003172 265 17 0 0 003131 FNEND JSP Q, OUTCHR OUTPUT CSS?
1998 003173 265 17 0 0 003146 JSP Q, INCHR INPUT LINE FEED
1999 003174 254 20 0 0 003166 JRST LINNUM GO TO LINNUM (SECTION B)
2000 003175 200 02 0 0 003101 RETURN MOVE B,A SAVE THE CHARACTER IN AC-A
2001 003176 201 01 0 0 003112 MOVE A."R" RETRIEVE A.DOC R
2002 003177 265 17 0 0 003179 JSP Q, OUTCHR OUTPUT RETURN STATEMENT CODE "R"
2003 003178 265 21 0 0 003166 MOVE A."R" CLRPRINT AC-A
2004 003179 254 00 0 0 003126 JRST +2
2005 003180 265 17 0 0 003176 JSP Q, INCHR INPUT CHARACTER
2006 003181 302 31 0 0 003111 CAIE A.11
2007 003182 308 01 0 0 003140 CAIE A.40
2008 003183 265 17 0 0 003125 JRST +2 IF CHARACTER IS A SPACE OR TAB, GO INPUT NEXT CHARACTER
2009 003184 322 01 0 0 003151 CAIE A.15 END OF LINE?
NO ERRORS DETECTED

PROGRAM BREAK IS 004863

SK CORE USED
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>INEND</td>
<td>574</td>
<td>599</td>
<td>673</td>
<td>700</td>
<td>769#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INFLP</td>
<td>102#</td>
<td>1122</td>
<td>1303</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPPN</td>
<td>1967#</td>
<td>1973</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>INPUT1</td>
<td>567#</td>
<td>596</td>
<td>670</td>
<td>698</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOD</td>
<td>182#</td>
<td>679</td>
<td>763</td>
<td>775</td>
<td>863</td>
<td>884</td>
<td>909</td>
<td>921</td>
<td>1004</td>
<td>1079</td>
<td>1101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOD8</td>
<td>184#</td>
<td>1976</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>IOT</td>
<td>162#</td>
<td>258</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>IP3</td>
<td>585</td>
<td>628#</td>
<td>651</td>
<td>669</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP4</td>
<td>627</td>
<td>705#</td>
<td>672</td>
<td>781#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPER2</td>
<td>577</td>
<td>596</td>
<td>672</td>
<td>781#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITI</td>
<td>313#</td>
<td>359</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>ITER1</td>
<td>239</td>
<td>274</td>
<td>321#</td>
<td>304</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITER2</td>
<td>294</td>
<td>325#</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>ITLEV</td>
<td>182#</td>
<td>312</td>
<td>315</td>
<td>361</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEVEL</td>
<td>181#</td>
<td>1613</td>
<td>1646</td>
<td>1686</td>
<td>1694</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LBRAC</td>
<td>1652</td>
<td>1658#</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>LINE</td>
<td>1944</td>
<td>238</td>
<td>1132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINO</td>
<td>1944</td>
<td>228</td>
<td>265</td>
<td>268</td>
<td>280</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINNUM</td>
<td>236#</td>
<td>278</td>
<td>295</td>
<td>367</td>
<td>371</td>
<td>375</td>
<td>435</td>
<td>440</td>
<td>462</td>
<td>466</td>
<td>470</td>
<td>545</td>
<td>549</td>
</tr>
<tr>
<td></td>
<td>553</td>
<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>LMDONE</td>
<td>314</td>
<td>354</td>
<td>362#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LMER1</td>
<td>332</td>
<td>338</td>
<td>348</td>
<td>362</td>
<td>364#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LMER2</td>
<td>338</td>
<td>358#</td>
<td>362#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LMER3</td>
<td>337</td>
<td>372#</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>LNM</td>
<td>239#</td>
<td>242</td>
<td>245</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPAR</td>
<td>1258</td>
<td>1281#</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>MEMERR</td>
<td>531</td>
<td>1424</td>
<td>1412#</td>
<td>1869</td>
<td>1917</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NIDX</td>
<td>635</td>
<td>674#</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>NDAND</td>
<td>166#</td>
<td>178#</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>NOTCNT</td>
<td>1782</td>
<td>1785#</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>NUCR1</td>
<td>256</td>
<td>262</td>
<td>275#</td>
<td>947</td>
<td>949</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUCR2</td>
<td>267</td>
<td>292#</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>NUMN</td>
<td>136#</td>
<td>1314#</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>NUMOUT</td>
<td>1783</td>
<td>1793#</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>DNED</td>
<td>859</td>
<td>877</td>
<td>883#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DNEDIM</td>
<td>537</td>
<td>554#</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>DPNR</td>
<td>1532</td>
<td>1535</td>
<td>1544#</td>
<td>1557</td>
<td>1562</td>
<td>1565</td>
<td>1574</td>
<td>1584</td>
<td>1592</td>
<td>1597</td>
<td>1601</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPNRD</td>
<td>1617</td>
<td>1619</td>
<td>1637#</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPTOR</td>
<td>1624</td>
<td>1642#</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>DSTACK</td>
<td>1634</td>
<td>1626</td>
<td>1637#</td>
<td>1655</td>
<td>1658</td>
<td>1665</td>
<td>1568</td>
<td>1691</td>
<td>1708</td>
<td>1718</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUT1</td>
<td>1772#</td>
<td>1789</td>
<td>1794</td>
<td>1806</td>
<td>1811</td>
<td>1826</td>
<td>349</td>
<td>357</td>
<td>380</td>
<td>343</td>
<td>461</td>
<td>544</td>
<td>567</td>
</tr>
<tr>
<td>DUTCHR</td>
<td>164#</td>
<td>229</td>
<td>223</td>
<td>278</td>
<td>273</td>
<td>367</td>
<td>664</td>
<td>667</td>
<td>687</td>
<td>692</td>
<td>709</td>
<td>718</td>
<td>728</td>
</tr>
<tr>
<td></td>
<td>752</td>
<td>765</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>777</td>
<td>801</td>
<td>804</td>
<td>818</td>
<td>828</td>
<td>865</td>
<td>866</td>
<td>895</td>
<td>898</td>
<td>911</td>
<td>923</td>
<td>941</td>
<td>964</td>
<td>966</td>
</tr>
<tr>
<td>979</td>
<td>996</td>
<td>997</td>
<td>998</td>
<td>1000</td>
<td>1002</td>
<td>1006</td>
<td>1025</td>
<td>1030</td>
<td>1032</td>
<td>1041</td>
<td>1081</td>
<td>1103</td>
<td>1118</td>
</tr>
</tbody>
</table>
**TITLE AUX4**

**COMMENT %**

**WRITTEN BY GARRET A. VANDER LUOT 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**

<table>
<thead>
<tr>
<th>SECTION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Inputs a character from the disk into AC-A on channel one.</td>
</tr>
<tr>
<td>B</td>
<td>Outputs a character from AC-A to the disk on channel two.</td>
</tr>
<tr>
<td>C</td>
<td>Outputs the octal number in AC-A to the disk.</td>
</tr>
<tr>
<td>D</td>
<td>Outputs the integer in AC-A to the disk.</td>
</tr>
<tr>
<td>E</td>
<td>Outputs the integer in AC-A to a user's terminal.</td>
</tr>
<tr>
<td>F</td>
<td>Types frame and line nos. in an error message.</td>
</tr>
<tr>
<td>G</td>
<td>Types the symbol in AC-G.</td>
</tr>
<tr>
<td>H</td>
<td>Inputs an integer from the disk into AC-B.</td>
</tr>
<tr>
<td>I</td>
<td>Inputs a floating point number from the disk into AC-B.</td>
</tr>
<tr>
<td>J</td>
<td>Outputs the floating point number in AC-A to the disk.</td>
</tr>
<tr>
<td>K</td>
<td>Inputs an integer from a user's terminal into AC-B.</td>
</tr>
</tbody>
</table>

**ENTRY POINTS TO AUX4.MAC**

<table>
<thead>
<tr>
<th>ENTRY POINT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLOUT</td>
<td>From CFRAM.MAC (sections L &amp; O), located in section J</td>
</tr>
<tr>
<td>FRMLIN</td>
<td>From PROS.MAC (sections D, E, F, G, H, I, J, K, L, M, P, V &amp; X) and CFRAM.MAC (all sections except A, N &amp; P), located in section F</td>
</tr>
<tr>
<td>FRMLIN2</td>
<td>From PROS.MAC (sections E, F, G, H, I, J, K, M &amp; BB), located in section F</td>
</tr>
<tr>
<td>II</td>
<td>From PROS.MAC (sections B, C, AA &amp; BB) and CFRAM.MAC (sections H &amp; I), located in section H</td>
</tr>
</tbody>
</table>
AUX4 MACRO 47(204)-7 09156 2-AUG-74 PAGE 1-2
AUX4 MAC 6-JUN-74

ENTRY FLOUT, INFLP, IT, IOD8
ENTRY INCHR, OUTCHR, IOD, IO1, FRMLIN, IT, IOD2, TYPSYM, FRMLINZ
EXTERN IBSF, OBUF, PG2, PDLST, ERRTOF, FRMNUM, LINE, PG3, EXPNT

117 000200  FLAGS=0
118 000201  A=1
119 000202  B=2
120 000203  C=3
121 000204  D=4
122 000205  E=5
123 000206  F=6
124 000207  G=7
125 000208  H=8
126 000209  I=9
127 000210  J=10
128 000211  K=11
129 000212  L=12
130 000213  M=13
131 000214  N=14
132 000215  O=15
133 000216  P=16
134 000217  Q=17
135 000218  R=18
136 000219  S=19
137 00021A  T=20
138 00021B  U=21
139 00021C  V=22
140 00021D  W=23
141 00021E  X=24
142 00021F  Y=25
143 000220  Z=26
144 000221  a=27
145 000222  b=28
146 000223  c=29
147 000224  d=30
148 000225  e=31
149 000226  f=32
150 000227  g=33
151 000228  h=34
152 000229  i=35
153 00022A  j=36
154 00022B  k=37
155 00022C  l=38
156 00022D  m=39
157 00022E  n=40
158 00022F  o=41
159 000230  p=42
160 000231  q=43
161 000232  r=44
162 000233  s=45
163 000234  t=46
164 000235  u=47
165 000236  v=48
166 000237  w=49
167 000238  x=50
168 000239  y=51
169 00023A  z=52
170 00023B  A=53
171 00023C  B=54
172 00023D  C=55
173 00023E  D=56
174 00023F  E=57
175 000240  F=58
176 000241  0=59
177 000242  1=60
178 000243  2=61
179 000244  3=62
180 000245  4=63
181 000246  5=64
182 000247  6=65
183 000248  7=66
184 000249  8=67
185 00024A  9=68
186 00024B  0=69
187 00024C  =70
188 00024D  ?=71
189 00024E  =72
190 00024F  =73
191 000250  =74
192 000251  =75
193 000252  =76
194 000253  =77
195 000254  =78
196 000255  =79
197 000256  =80
198 000257  =81
199 000258  =82
200 000259  =83
201 00025A  =84
202 00025B  =85
203 00025C  =86
204 00025D  =87
205 00025E  =88
206 00025F  =89

000001 44 46 60 00 000077  POINT: POINT 6,G

133 002020  INCHR: SODGE IBSF+2; DECREMENT CHARACTER COUNT
134 002021  254 00 00 00 002005;  JRST GETBUF; IF BUFFER IS EMPTY, GO FILL IT
135 002022  134 00 00 00 002001;  IOD8 A; IBSF+1; PUT CHARACTER IN AC-A
136 002023  020225;  17 002000;  JRST (Q); YES, RETURN
137 002024  020205;  254 00 00 002000;  GETBUF: IN I; IF FULL BUFFER
138 002025  020206;  254 00 00 002001;  JRST INCHR: IF NO ERROR, GO GET NEXT CHARACTER
139 002026  020207;  020205;  STATE 1; EOF; END OF FILE?
140 002027  020210;  254 00 00 002000;  JRST (Q); RETURN
141 002028  DISK INPUT ERROR ON CHANNEL 1 = AUX4
142 002029  DC3 3, EASCZ /
143 00202A  051 03 00 002448;  1-J
144 00202B  051 34 00 002450;  HALT

151 000013  OUTCHR: SODGE OBUF+2; DECREMENT CHARACTER COUNT
152 000014  254 00 00 002017;  JRST PUTBUF; IF BUFFER IS FULL, GO EMPTY IT
153 000015  134 00 00 002001;  IOD8 A; OBUF+1; PUT CHARACTER IN BUFFER
154 000016  020210;  17 002000;  JRST (Q); RETURN
155 000017  020217;  254 00 00 002000;  PUTBUF: OUT 2; IF BUFFER IS EMPTY BUFFER
156 000018  020202;  254 00 00 002013;  JRST OUTCHR; IF NO ERROR, GO PUT CHARACTER IN BUFFER
157 000019  DISK OUTPUT ERROR ON CHANNEL 2 = AUX4
158 00001A  DC3 3, EASCZ /
159 00001B  051 03 00 002451;  1-J

SECTION A. THIS SUBROUTINE INPUTS A CHARACTER FROM THE DISK
ION CHANNEL 1. THE CHARACTER IS PLACED IN AC-A.

SECTION B. THIS SUBROUTINE OUTPUTS A CHARACTER TO THE DISK
ION CHANNEL 2. THE CHARACTER MUST BE IN AC-A.
SECTION F. THIS SUBROUTINE TYPES THE FRAME AND LINE NUMBER IN WHICH AN ERROR WAS DETECTED.

`FRMLN2: MOVEM Q,PC3 ;SAVE RETURN ADDRESS
AOS ERRTOT ;INCREMENT ERROR TOTAL
FRMLN: MOVEM Q,PC3 ;SAVE RETURN ADDRESS
AOS ERRTOT ;INCREMENT ERROR TOTAL
CIN A,12 ;END OF LINE?
JNST EL 1 IF END OF FRAME, GO TO EL
JSP Q,INCHR ;INPUT CHARACTER
CALL A,12 ;END OF LINE?
JNST ,2
JSP Q,OUTCHR ;OUTPUT <15>
JSHOP 3,ASCII /
FRAME */
MOVE A,FRMNUM ;GET THE CURRENT FRAME NUMBER
JSP 1,10T ;TYPE THE FRAME NO.
TTCALL 3,ASCII / ;LINE */
MOVE A,LINE ;GET THE CURRENT LINE NUMBER
JSP 1,10T ;TYPE THE LINE NO.
JNST PPC3 ;RETURN

SECTION G. THIS SUBROUTINE TYPES ILLEGAL SYMBOLS.

`FRMTYP: MOVEM Q,PC2 ;SAVE RETURN ADDRESS
TTCALL 3,ASCII / ;/1
MOVE Q,6
MOVE B,POINT ;PUT BYTE POINTER IN AC-B
TSI ;LOAD A,B
ADD ;A,40 ;CONVERT FROM SIXBIT TO ASCII
TTCALL 1,A ;TYPE THE CHARACTER
JSP Q,OUTCHR ;OUTPUT CHARACTER
JST PPC2 ;RETURN

SECTION H. THIS SUBROUTINE TYPES AN INTEGER FROM THE DISK AND PLACES IT IN AC-B.

`INTYP: MOVEM Q,PC2 ;SAVE RETURN ADDRESS
SETZ B,
JSP Q,INCHR ;INPUT CHARACTER
AOS F ;INCREMENT LESSON CHARACTER COUNT
SECTION 1, THIS SUBROUTINE INPUTS A FLOATING POINT NUMBER FROM THE DISK AND PLACES IT IN AC-B.
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
SECTIN K, THIS SUBROUTINE INPUTS AN INTEGER FROM A USER'S TERMINAL AND PLACES IT IN AC-B.

J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

023424 403 32 0 00 023200 IITI SETE B, TDCALL 4,A BUTPUT ASCII CHARACTER 023426 301 21 0 00 023425 CAN A,40 IS CHARACTER A SPACE? 023427 254 02 0 00 023425 JSTI +,2 YES, GO INPUT NEXT CHARACTER 023431 254 32 0 00 023432 JSTI +,2 023432 251 24 0 00 023421 IIT2 TDCALL 4,A INPUT NEXT CHARACTER 023432 301 21 0 00 023425 CAN A,60 023431 123 21 0 00 023421 IIT3 CALA A,60 023434 254 22 0 07 023430 JSTI (Q) RETURN IF CHARACTER IN NOT A DIGIT 023435 221 22 0 02 003012 IMULI B,12 MULTIPLY BY 12 003436 271 02 0 01 0077720 ADDI B,-60(A) ADD DIGIT TO NO, ALREADY IN AC-B 023437 254 00 0 00 023431 JSTI IIT1 GO GET NEXT DIGIT NO SYM END

PROGRAM BREAK IS 003472

3K CORE USED
<table>
<thead>
<tr>
<th></th>
<th>101</th>
<th>129</th>
<th>213</th>
<th>232</th>
<th>265</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>
</tbody>
</table>

- Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
TITLE PRSLOW

COMMENT %

WRITTEN BY GARRET A. VANDER LUGT COMPLETED 6-JUN-74

LOW SEGMENT FOR LPROS, SHR, VERSION 3

PRSLOW, 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 SSAE MONITOR COMMAND.

ENTRY INBUF, OUTBUF, ARGNT, NUMST

INTERN PSTACK, PTYPE, HOLD, PC5, EXPNT, RSNUM, TRMBLK, JOBNO,
INTERN PC4, ADDR, SYMIDX, SYMTAB, SSTACK, STYPE, OSTACK, ASGN, LEVEL, ITRLEV
INTERN PART, LINE, FRMTOT, ERRTOT, FRMNUM, FRMTYP
INTERN STORE, INAM, ONAM, PC1, IBUF, OBUF, PC2, PDLST, PPN, PC3
INTERN DELS, PRST, PRSFN, RFMCNT, LINENO, GOTO

000137

LOC JOBVER

XAD 301, 1

PELOC 9

STORE: BLOCK 1 I USED BY LPROS, MAC

ARGNT: BLOCK 6 I USED BY LPROS, MAC

JOBNO: BLOCK 1 I JOB NUMBER

TRMBLK: BLOCK 3 I TRMOP, ARGUMENTS

INTBLK: BLOCK 5

INTLOC: BLOCK 12

IBUF: BLOCK 3 I USED BY AUX4, MAC

OBUF: BLOCK 3 I USED BY AUX4, MAC

ONAM: BLOCK 4 I USED BY PROS, MAC

PPN: BLOCK 1

INBUF: BLOCK 203

OUTBUF: BLOCK 203

ANY CHANGES MADE ABOVE THIS POINT MUST ALSO BE MADE IN

ILOW, MAC, LDTLOW, MAC, AND RPTLOW, MAC,

PC1: BLOCK 1 I USED BY PROS, MAC

PC2: BLOCK 1 I USED BY PROS, MAC AND AUX4, MAC

PC3: BLOCK 1 I USED BY PROS, MAC, CFROM, MAC, AND AUX4, MAC

PC4: BLOCK 1 I USED BY CFROM, MAC

PC5: BLOCK 1 I USED BY CFROM, MAC

PDLST: BLOCK 10 I USED BY AUX4, MAC
"LEARN LESSON LISTING"

LESSON: HELP
DATE: 8/2/74
TIME: 1140
PPN: 24805, 24893

1 5 TYPE-O
2 TEXT
3 THIS IS THE LEARN HELP LESSON. WHAT TERMINAL ARE YOU USING?\-2
4 1) TELETYPE
5 2) TEK 4813
6 3) TEK 4902
7 \ ANSWERS
8 1 0 1
9 2 0 2
10 3 0 3
11 ? 0
12 ACTIONS
13 2 3
14 CEO-8A-1
15 0
16 RIPLEASE TYPE 1, 2, OR 3.\n17

1 10 TYPE-O
2 PAGE
3 TEXT
4 YOU HAVE THE FOLLOWING OPTIONS, \-2
5 A) A GENERAL DESCRIPTION OF LEARN
6 B) A LIST OF LEARN SYSTEM COMMANDS
7 C) A DESCRIPTION OF LEARN EDIT COMMANDS
8 D) A DESCRIPTION OF THE LEARN INSTRUCTIONAL LANGUAGE
9 E) A DESCRIPTION OF THE CALCULATION MODE
10 F) TERMINATE THE HELP LESSON\-2
11 TYPE THE LETTER INDICATING YOUR CHOICE:
12 \ ANSWERS
13 EXTRA OFF
14 \ A 0 A
15 B 0 B
16 C 0 C
17 D 0 D
18 E 0 E
19 F 0 F
20 \ ACTIONS
21 A 0
22 B 0
23 \
REPRODUCED WITH PERMISSION OF THE COPYRIGHT OWNER. FURTHER REPRODUCTION PROHIBITED WITHOUT PERMISSION.

PAGE 3

42
43 ? 0
44 ACTIONS
45 A
46 B10
47 B
48 E
49 Y
50 \R PLEASE ANSWER YES OR NO.

32 TYPE-G

PAGE

TEXT

THE FOLLOWING COMMANDS MAY BE TYPED AFTER "ENTER COMMAND": \n
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
RENAME - 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.

40 TYPE-G

ANSWERS
EXTRA OFF
BYE CLEAR COPY DELETE EDIT FILES GET HELP JOB LIST MONITOR PROCESS
16. M 0 RENAME
17. N 0 REPORT
18. O 0 TEST
19. P 0 UPDATE
20. Q 0 WRITE
21. X 0 X
22. E 0 COMMANDS
23. T 0 ACTIONS
24. A
25. B 198
26. B 1180
27. B 1180
28. C
29. D 1180
30. D 1180
31. E 1180
32. E 1180
33. F 1180
34. F 1180
35. G 1180
36. G 1180
37. H 1180
38. H 1180
39. I
40. J 1180
41. J 1180
42. K 1180
43. K 1180
44. L 1180
45. L 1180
46. M 1180
47. M 1180
48. N 1180
49. N 1180
50. O 1180
51. O 1180
52. P 1180
53. P 1180
54. Q 1180
55. Q 1180
56. R 1180
57. R 1180
58. S
59. S
60. T 1180
61. T 1180
62. U
63. V
64. W
65. X
66. Y
67. Z

PLEASE TYPE A COMMAND NAME, OR TYPE X, N

1. 50 TYPE-Q
2. PAGE
3. TEXT
4. DO YOU WANT ANY MORE INFORMATION?
00

D

:::

~~=~ ::L;m;;;•·~

~~.~m1~1~J~:

0

ll>:;

-~~

!i~

Iii~

zz oc

~~

.z-

iE~

;j

~:~f~!_:l

148x232

E

u X

z

Ia

~~5~

::::.:m:•E

139x144

I~1e~m~~~~~~~

246x245

I

225x242

._.. gj

242x233

~

243x230

xi

225x223

:IU

235x223

& gl

243x219

WI

242x214

ii

243x205

~

225x190

~a0 ~~

225x171

~N~~.1~e~~

238x171

N=l

::;;;:

251x147

~

256x147

..

217x137

~N~~.1~e~~
100 TYPE-O
 PAGE
 TEXT
 THE CLEAR COMMAND IS USED TO INITIALIZE THE TRANSACTION
 FILE AND ANSWER FILES FOR STORING STUDENT PERFORMANCE DATA.\
 THE FORM OF THE CLEAR COMMAND IS:\-2
 CLEAR <LESSON NAME>\-2
 WHERE:\-2
 <LESSON NAME> IS THE FILENAME OF THE LESSON FOR WHICH ANSWERS
 WILL BE STORED\-2
 IF <LESSON NAME> IS OMITTED, THIS COMMAND CLEARS THE TRANSACTION FILE.
 RECORD FILES MUST BE ESTABLISHED USING THE CLEAR COMMAND IN ORDER
 FOR STUDENT PERFORMANCE DATA TO BE STORED. RECORD FILES SHOULD BE
 ESTABLISHED ONLY UNDER THE INSTRUCTOR'S IDENTICAL PROJECT.
 PROGRAMMER NOS. MOST STUDENT PERFORMANCE DATA IS STORED IN THE
 TRANSACTION FILE; HOWEVER, ACTUAL ANSWER STRINGS ARE STORED IN
 THE ANSWER FILE WITH THE SAME NAME AS THE LESSON WHICH IS BEING
 EXECUTED.\-2
 EXAMPLES: CLEAR
 \-2
 ACTIONS
 CLEAR LESSN

110 TYPE-O
 PAGE
 TEXT
 THE COPY COMMAND IS USED TO MAKE COPIES OF LEARN FILES.
 ONLY FILES IN THE USER'S OWN DISK AREA MAY BE COPIED.\-2
 THE FORM OF THE COPY COMMAND IS:\-2
 COPY <NEW NAME> <OLD NAME> <SWITCH>\-2
 WHERE:\-2
 <NEW NAME> IS THE NAME GIVEN TO THE NEW FILE
 <OLD NAME> IS THE NAME OF THE FILE BEING COPIED
 <SWITCH> IS AN OPTIONAL ARGUMENT THAT INDICATES WHICH
 KIND OF FILE IS BEING COPIED
 U - FOR AN UNPROCESSED LESSON FILE
 P - FOR A PROCESSED LESSON FILE
 R - FOR A RECORD FILE
 A - FOR AN ANSWER FILE\-2
 IF <SWITCH> IS OMITTED, AN UNPROCESSED FILE WILL BE COPIED.\-2
 EXAMPLES: COPY TEMP=LESSN/U
 \-2
 ACTIONS
 COPY TEMP=MASTER/U

120 TYPE-O
 PAGE
 TEXT
 THE DELETE COMMAND IS USED TO DELETE LEARN FILES FROM THE
 DISK OR TO DELETE INFORMATION FROM THE MASTER RECORD FILE.\-2
THE FIRST FORM OF THE DELETE COMMAND IS: \( \text{DELETE <FILENAME>/<SWITCH>\#2} \)

- **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
  - **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`

THE SECOND FORM OF THE DELETE COMMAND IS: \( \text{DELETE <LESSON>/<PPN1>/<PPN2>/<DI>/<DO>/<DF1>/<DF2>/I/\#2} \)

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 program nos. are in the range **<PPN1>-<PPN2>** is to be deleted (also optional).
- **<DI>** and **<DO>** specify that information gathered between the dates **<DI>-<DO>** is to be deleted (optional).
- **<DF1>** and **<DF2>** specify that information on frames **<DF1>-<DF2>** is to be deleted (optional)

THE **/I** specifies that information is to be deleted from the master record file: master, rec.

**EXAMPLES:**
- `DELETE LESSN1/I`
- `DELETE 2488-2430/I`
- `DELETE LESSN1,2488-2410/I`
- `DELETE LESSN2,2430,12/12/74-3/12/74/I`

**ACTIONS**
- `BIBB`

---

138 TYPE-O

**PAGE**

**TEXT**

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:

\( \text{EDIT <LESSON NAME>/<INCREMENT>\#2} \)

WHERE:
- **<LESSON NAME>** 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.
I

EXAMPLE 1: EDIT LESSON 3

16
17
18
19

150 TYPE-Q

EXAMPLES

140 TYPE-Q

PAGE

ACTION

TEXT

B100

THE FILES COMMAND IS USED TO OBTAIN A LISTING OF LEARN
FILES WHICH ARE STORED IN THE USER'S DISK AREA.

WHERE:

- <SWITCH> IS AN OPTIONAL ARGUMENT THAT INDICATES THE
  KINDS OF FILES TO BE LISTED
  U - FOR UNPROCESSED LESSON FILES
  P - FOR PROCESSED LESSON FILES
  R - FOR RECORD FILES
  A - FOR ANSWER FILES

IF <SWITCH> IS OMITTED, ALL FILES WILL BE LISTED. MORE THAN
ONE LETTER MAY BE USED FOR THE SWITCH.

150 TYPE-Q

PAGE

TEXT

B100

THE GET COMMAND IS USED BY STUDENTS TO RUN A LESSON WHICH
HAS BEEN PLACED ON THE COMPUTER BY THEIR INSTRUCTOR.

WHERE:

- <LESSON NAME> IS THE FILENAME WHICH HAS BEEN ASSIGNED
  TO THE LESSON
- <PPN> IS AN OPTIONAL ARGUMENT WHICH CONSISTS OF THE
  PROJECT-PROGRAMMER NOS., SEPARATED BY A COMMA, UNDER
  WHICH THE LESSON IS STORED

IF <PPN> IS OMITTED, LEARN SEARCHES FOR THE LESSON UNDER
IDENTICAL PROJECT-PROGRAMMER NOS. WHICH ARE EQUAL TO THE USER'S
PROJECT NO. FOR THIS REASON, INSTRUCTORS SHOULD NORMALLY HAVE
IDENTICAL PROJECT-PROGRAMMER NOS., AND STUDENT PROJECT
NOS., SHOULD BE THE SAME AS THEIR INSTRUCTOR'S PROJECT NO.

EXAMPLES:

GET LESSN

GET LESSN (24000, 24005)

EXAMPLES:

GET LESSN

B100
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.

170 TYPE-Q
PAGE
TEXT
THE KJOB COMMAND IS USED TO INITIATE THE LOGOUT PROCESS.
This command is equivalent to the the KJOB MONITOR command.
KJOB
There are no arguments for this command.

180 TYPE-Q
PAGE
TEXT
THE LIST COMMAND IS USED TO OBTAIN LESSON LISTINGS ON THE
LINE PRINTER.
THE FORM OF THE LIST COMMAND IS:
LIST LESSON/NO/COPIES
WHERE LESSON is the filename which has been assigned
TO THE LESSON
COPIES is an optional argument which indicates the
NUMBER OF COPIES THAT WILL BE PRINTED.
EXAMPLE: LIST LESSON/C

190 TYPE-Q
PAGE
TEXT
THE MONITOR COMMAND IS USED TO EXIT FROM LEARN AND RETURN TO
MONITOR LEVEL.
THE FORM OF THE MONITOR COMMAND IS:
MONITOR
There are no arguments for this command.

200 TYPE-Q
PAGE
TEXT
THE MONITOR COMMAND IS USED TO EXIT FROM LEARN AND RETURN TO
MONITOR LEVEL.
THE FORM OF THE MONITOR COMMAND IS:
MONITOR
There are no arguments for this command.
**200 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:\n
```
PROCESS <LESSON NAME> <F1> <F2>
```

WHERE:\n
- `<LESSON NAME>` IS THE FILENAME OF THE LESSON TO BE PROCESSED
- `<F1>` AND `<F2>` ARE OPTIONAL ARGUMENTS THAT INDICATE THAT
  AI ALL FRAMES WITH FRAME NOS. BETWEEN F1 AND F2 ARE TO BE
  PROCESSED.

IF `<F1>` AND `<F2>` ARE OMITTED, THE ENTIRE LESSON WILL BE

**EXAMPLES:**

- PROCESS LESSN
- PROCESS LESSN1/100
- PROCESS LESSN4/200-400

**ACTIONS**

**B180**

---

**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:\n
```
RENAME <NEW NAME> <OLD NAME> <SWITCH>
```

WHERE:\n
- `<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 PHYS4.100
- RENAME ANSAMP/MASTER/R

**ACTIONS**

**B180**

---

**220 TYPE-Q**

**PAGE**

**TEXT**

THE REPORT COMMAND IS USED TO OBTAIN LEARN REPORTS.

---
THE FORM OF THE REPORT COMMAND IS \*2

REPORT <DEVR> <LESSNR>, <F1> <F2> \* (E1 <E2>) \* (F1 <F2>) \* (S) \*2

WHERE \*2

<DEVR> INDICATES THE DEVICE ON WHICH THE REPORT IS TO BE
PRINTED AND THE STYLE OF THE REPORT. <DEVR> MAY BE TTY
(TTYPPED), TTY (SUMMARY REPORT ON LINE PRINTER), OR
LPT (LINE PRINTER - DETAILED REPORT).

<LESSNR> IS THE FILENAME OF THE LESSON FOR WHICH A REPORT
IS DESIRED.

<F1> AND <F2> ARE PROGRAMMER NOs. OF STUDENTS.

<E1> AND <E2> 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. \*2

ALL OF THE ARGUMENTS ARE OPTIONAL. \*2

EXAMPLES

REPORT TTYLNE1
REPORT LPTILNE2
REPORT TTYLNE3/S
REPORT TTY240B=240B/S
REPORT LPTILNE3=240B=240B=240B=240B
REPORT TTYLNE5=2/25/74=5/25/74
REPORT TTYLNE1

ACTIONS

B198

230 TYPE-O

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. \*2

THE FORM OF THE TEST COMMAND IS \*2

TEST \* LESSON NAME >\* <F1> <F2> = <F2> \*2

WHERE \*2

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) \*2

THE LESSON IS PROCESSED BEFORE EXECUTION IF THE PROCESSED

LESSON FILE IS OLDER THAN THE UNPROCESSSED FILE. THE FOLLOWING

TEST MODE COMMANDS MAY BE TYPED WHEN LEARN IS WAITING FOR

AN ANSWER TO ANY OF THE QUESTIONS IN THE LESSON1

IN = BRANCH TO FRAME N (E.g., IN 0, 100)

IE = ENTER THE LESSON EDITOR

T = TERMINATE THE LESSON \*2

EXAMPLES: TEST LNE1

TEST LNE2/150-250

ACTIONS

B198

240 TYPE-O
THE WRITE COMMAND IS USED TO ENTER A NEW LESSON INTO
THE COMPUTER SYSTEM.\textbackslash{}

THE FORM OF THE WRITE COMMAND IS:\textbackslash\textbackslash{}

WRITE LESSON NAME/INCERTEN/\textbackslash\textbackslash{}

WHERE\textbackslash\textbackslash{}

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.\textbackslash\textbackslash{}

IF <INCREMENT> IS OMITTED, THE FRAME INCREMENT WILL BE 10.\textbackslash\textbackslash{}

EXAMPLE: WRITE LESS1/5

250 TYPE-Q

THE UPDATE COMMAND IS USED TO TRANSFER INFORMATION FROM THE
TRANSACTION FILE TO THE MASTER RECORD FILE.\textbackslash\textbackslash{}

THE FORM OF THE UPDATE COMMAND IS:\textbackslash\textbackslash{}

UPDATE<\textbackslash\textbackslash{}

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:\textbackslash\textbackslash{}

1) A - APPEND FRAMES ONTO END OF LESSON
2) AN - APPEND LINES ONTO FRAME N\textbackslash\textbackslash{}
3) CN - COPY FRAME N, AND NUMBER IT N\textbackslash\textbackslash{}
4) DN - DELETE FRAME N\textbackslash\textbackslash{}
5) Dn - DELETE FRAMES N THRU H\textbackslash\textbackslash{}
6) DN,P - DELETE LINE P IN FRAME N\textbackslash\textbackslash{}
7) DN,P,Q - DELETE LINES P THRU Q IN FRAME N\textbackslash\textbackslash{}
8) H - TYPE THIS SUMMARY\textbackslash\textbackslash{}
9) IN - INSERT FRAME N\textbackslash\textbackslash{}
10) IN,P - INSERT IN FRONT OF LINE P IN FRAME N\textbackslash\textbackslash{}
11) KN - ADD N TO ALL FRAME NOS.\textbackslash\textbackslash{}
12) KN,H - ADD N TO ALL FRAME NOS, STARTING
   AT FRAME H\textbackslash\textbackslash{}
13) NN,N - RENUMBER FRAME N TO N\textbackslash\textbackslash{}
14) RN - REPLACE FRAME N\textbackslash\textbackslash{}
15) RN,P - REPLACE LINE P IN FRAME N\textbackslash\textbackslash{}
16) RN,P,Q - REPLACE LINES P THRU Q IN FRAME N
22 4) RN,P,OLD-STRING,NEW-STRING - REPLACE OLD-STRING IN LINE P FRAME N\#2
23 2) T - TYPE THE ENTIRE LESSON
24 2) TN - TYPE FRAME N
25 3) TN-M - TYPE FRAMES N THRU M
26 4) TN,P - TYPE LINE P IN FRAME N
27 5) TN,P-Q - TYPE LINES P THRU Q IN FRAME N\#2
28 1) XN - CHANGE FRAME INCREMENT TO N\#2
29 1) EP - EXIT FROM THE EDITOR
30 2) ET - EXIT AND PROCESS LESSON
31 3) ET - EXIT, PROCESS, AND TEST LESSON\#2
32 1) M - RETURN TO MONITOR LEVEL

33 \ 
34 ACTIONS
35 B150

1 276 TYPE-Q
2 PAGE
3 TEXT
4 THE CALCULATION MODE ALLOWS STUDENTS AND LESSON AUTHOR'S 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 CALCU-
11 LATION MODE.\#2
12 THE FOLLOWING MATHEMATICAL OPERATIONS MAY BE PERFORMED: \#2
13 ADDITION -
14 SUBTRACTION -
15 MULTIPLICATION -
16 DIVISION -
17 EXPONENTIATION \#2
18 THE FOLLOWING FUNCTIONS ARE ALSO AVAILABLE: \#2
19 ABSOLUTE VALUE
20 INT - INTEGER
21 SQRT - SQUARE ROOT
22 ABSRT - CUBE ROOT
23 LOG - COMMON LOGARITHM
24 NLOG - NATURAL LOGARITHM
25 EXP - EXPONENTIAL
26 SIN - SINE (RADIAN)
27 COS - COSINE (RADIAN)
28 COSD - COSINE (DEGREES)
29 TAN - TANGENT (RADIAN)
30 TAND - TANGENT (DEGREES)
31 ASIN - INVERSE SINE (RADIAN)
32 ACOS - INVERSE COSINE (RADIAN)
33 ASIND - INVERSE SINE (DEGREES)
34 ACOSD - INVERSE COSINE (DEGREES)
35 ATAN - INVERSE TANGENT (RADIAN)
36 ATAND - INVERSE TANGENT (DEGREES) \#2
37 THE MOST COMMONLY USED CALCULATION MODE STATEMENT IS THE "TYPE"
STATEMENT. THE "TYPE" STATEMENT INSTRUCTS LEARN TO TYPE THE
RESULTS OF CALCULATIONS. HERE ARE SOME EXAMPLES:

- \[ \text{TYPE } 3^{4/2.56} \]
- \[ \text{TYPE } \sqrt{256} \]
- \[ \text{TYPE } 3^{4/(234+47)} \]
- \[ \text{TYPE } \log_{10}(34) \]

NUMBERS MAY BE WRITTEN IN EXPONENTIAL NOTATION, THE POWER OF
TEN IS WRITTEN AFTER THE LETTER "E" AS SHOWN IN THE
EXAMPLES BELOW:

- \[ 2.5 \times 10^2 \text{ (2.5 TIMES 10 TO THE 2ND POWER)} \]
- \[ 1.7 \times 10^{-5} \text{ (1.7 TIMES 10 TO THE MINUS 5TH POWER)} \]

SEE THE "LEARN LESSON AUTHOR'S GUIDE" FOR MORE ADVANCED
CALCULATION MODE FEATURES.

WOULD YOU LIKE TO WORK SOME EXAMPLE PROBLEMS?

\[ \]

\[ \text{ANSWER} \]

\[ \text{A) YES} \]

\[ \text{B) NO} \]

\[ \text{C) OTHER} \]

\[ \]

\[ \text{REMEMBER TO TYPE THE "ESC" (OR ALT MODE) KEY TO GET} \]
\[ \text{INTO THE CALCULATION MODE, THEN TYPE} \]
\[ \text{TYPE } 3479+345 \]
\[ \text{TO COMPUTE THE ANSWER,} \]
\[ \text{FINALLY, TYPE THE "ESC" (OR ALT MODE) KEY TO} \]
\[ \text{GET OUT OF CALCULATION MODE, NOW GO AHEAD AND DO THE} \]
\[ \text{PROBLEM.} \]

\[ \]

\[ \text{ANSWER} \]

\[ 1 \times 3479 \times 345 \]

\[ \]

\[ \text{ACTIONS} \]

\[ F1 \]

\[ ? \]

\[ RI \]

\[ P \]

\[ F1 \]

\[ A+1 \]

\[ \]

\[ \text{WHAT IS THE SQUARE ROOT (\text{Sqrt}) OF 234?} \]

\[ \]

\[ \text{TYPE-0} \]

\[ \text{PAGE} \]

\[ \text{TEXT} \]

\[ \]
\ ANSWER.
1 + SQRT(256)
* ACTION
1

Fi
?
R1
*
Fi
A1

302 TYPE-Q
PAGE
TEXT
HOW MUCH IS 5.6E24 DIVIDED BY THE QUANTITY 2000 TIMES 37.257
\ ANSWER
\ 5.6E24/(2000*37.25)
2 = 5.6E24/2000*37.25
* ACTION
Fi
B150
2
Fi

11  / THE PROPER CALC MODE STATEMENT IS
12  TYPE 5.6E24/(2000*37.25)
13  YOU FORGOT TO USE PARENTHESES. TRY AGAIN.
14
15
16
B150

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