8-1977

Development and Application of Computer Assisted Instruction for Mini-Computers

Thomas A. Burne
Western Michigan University

DEVELOPMENT AND APPLICATION OF COMPUTER ASSISTED INSTRUCTION FOR MINI-COMPUTERS

by

Thomas A. Burne

A Thesis
Submitted to the
Faculty of The Graduate College
in partial fulfillment
of the
Degree of Master of Arts

Western Michigan University
Kalamazoo, Michigan
August 1977

ACKNOWLEDGEMENTS

I would first like to express my sincere appreciation to Dr. Arthur Snapper for both the opportunity to work in his laboratory and for his guidance and support.

I would like to thank my wife, Nancy, for her encouragement during the course of my studies and for her help in the editing and typing of this manuscript.

Finally, I would like to thank Dr. Douglas Anger who initiated and cultivated my interest in the study of the experimental analysis of behavior, and who helped to shape my goals through his friendship.

Thomas A. Burne

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

MASTERS THESIS 13-10,388

BURNE, Thomas Alan
DEVELOPMENT AND APPLICATION OF
COMPUTER ASSISTED INSTRUCTION FOR
MINI-COMPUTERS.

Western Michigan University, M.A., 1977
Psychology, experimental

Xerox University Microfilms, Ann Arbor, Michigan 48106

TABLE OF CONTENTS

INTRODUCTION

Traditional educational techniques show little evidence of respect for or utilization of the results of research in the field of learning. Education, the branch of technology which is most concerned with the learning process, does not appear to observe the basic principles of learning developed through research in the experimental analysis of behavior.

Three notable shortcomings in traditional educational technology are especially evident. First, in most schools today, the contingencies in effect are not based on positive control, but, in fact, fall under the area of aversive control. Skinner (1959) pointed out that the student behaves primarily to escape the threat of a series of minor aversive events, such as instructor disapproval, peer criticism, etc. Arriving at the correct answer is in itself an insignificant event amid the inevitable by-products of aversive control, such as boredom, anxiety, and aggression. Secondly, the contingencies of reinforcement utilized are far from optimal. Unless explicit mediating behavior is determined in advance, the lapse of a few seconds between the response and reinforcement will destroy the effect. Traditionally, exam feedback is delayed days to weeks, in light of which it is extremely surprising to note that this system has had any effect at all. A third shortcoming is the lack of an efficient shaping system to develop the desired response through a series of short steps, each systematically reinforced.

The experimental analysis of behavior is concerned with contingencies which control behavior. The major components of this are the situation in which the behavior occurs, the behavior itself, and the consequences of the behavior (Skinner, 1959). The majority of these principles and techniques have been developed in the animal laboratory. In the laboratory, the environment can be regulated to optimize the desired conditions. Such factors as deprivation and available stimuli can all be regulated by the experimentor. The response, i.e., the behavior itself, can also be easily defined, manipulated, and detected by the experimentor. Finally, in the animal laboratory, the programming of the consequences of behavior can be done with great accuracy.

Through manipulation of these contingencies, the animal can be taught to discriminate. A discrimination occurs when an animal has been trained to make one response in the presence of one stimulus, and another response in the presence of a different stimulus. This discriminative performance in the animal laboratory can be modified by manipulating the reinforcement contingencies associated with each stimulus.

Michael (1974) pointed out that the essential contingencies in effect in the animal laboratory are also in effect in human learning, i.e., acquiring verbal behavior. In most courses, what the student learns is to emit one response in the presence of one stimulus, and another response in the presence of a different stimulus. Michael noted that college learning in an animal training paradigm would be a form of multiple-stimulus, multiple-response discrimination learning.

However, in human learning, the three components of animal learning are not as simple to identify. The environment is not controlled to as great an extent as it is in animal studies. The animal establishes a repertoire by emitting responses to environmental stimuli, followed by experimentor arranged consequences contingent upon performance. In human learning, this sequence is not as easily controlled. Defining and consequating a response is not a simple, one-step process. A student makes a response when reading a text. What are the the consequences of the behavior? The student may listen to lectures and read textbooks, but the reinforcement for such behavior is not as direct as is the pellet of food used to reinforce a hungry animal. It has been said that students attend college (a response) to receive an education (a reinforcer). However, it is in the steps taken to obtain that education that the principles of operant behavior and task analysis come into effect.

The concept of task analysis has emerged in response to the pressures of practical instructional design. Task analysis examines the properties of a behavior which is to be learned and the sequencing of the component tasks. Task analysis first identifies the necessary prerequisite skills which the learner must possess before being able to successfully complete the task or objective (Bunderson, 1970). This sequence is repeated for each task or objective.

Michael (1974) summarized the differences between the animal laboratory and the most common form of college learning, i.e., studying a text, and found two major differences. First of all, Skinner's stimulus-response-consequence learning formula is difficult to apply. It is simple in the animal laboratory to define performance in terms of stimuli, responses, and their consequences. However, these elements do not appear to exist in any obvious form in the textbook learning situation. Secondly, the learner has control over the presentation of the stimulus material in the college learning situation, whereas this is not the case in the animal laboratory.

A number of educational theories and systems have been developed based upon Skinner's analysis of the pertinent variables necessary for learning. The first of these was Skinner's teaching machine approach. From Skinner's point of view, education was still based on aversive control, the by-products of which were early drop-out, mental fatigue, forgetfulness, and apathy. In spite of progressive trends to utilize available reinforcers, Skinner felt that the bulk of educational technology was not as effective as possible. Conditioned reinforcers, such as grades and diplomas, were not directly contingent on behavior in an effective manner. Skinner stated that to properly arrange the number of contingencies which could be used in a single experimental session would require some type of mechanical help. The human organism is equally, if not more sensitive to the precise contingencies studied in the animal laboratory. To program these contingencies would require instrumental assistance.

Skinner's early teaching machine, as described in Cumulative Record (1959), consisted of a box with a window through which the student could read a question or problem printed on paper. In order to respond, the student set a numbered slide to the appropriate answer choice and turned a knob on the side of the machine. Reinforcement was immediate because the knob turned only if the attempted answer was right. Wrong answers were counted, but had no effect. One of the advantages of Skinner's teaching machine was that the student had to pass through a carefully designed sequence of steps in acquiring a complex behavior. Each step was small, and moved the student closer to the designated behavior. A disadvantage to this early machine was that it could handle only multiple choice questions, thereby requiring no originality on the part of the student. Skinner later developed another teaching machine which allowed the student to write in the answer.

Skinner indicated several ways in which a teaching machine can be effective in much the same way as a one-to-one private tutor. First of all, there is constant interaction between the program and the student. Unlike textbooks, this induces sustained activity. Secondly, the machine requires complete comprehension of every point, either frame by frame or lesson by lesson, before the student can continue. Texts, on the other hand, proceed without this, and can leave a student behind. Thirdly, sequencing presents only that material which the student is ready for, is most likely to understand, and can respond to correctly. But the most effective aspect of the teaching machine is its immediate reinforcement of each correct response in order to efficiently shape the targeted behavior.

The next type of educational technology to be developed was the contingency managed approach, which sprang from Keller's (1968) lead. "Contingency is a relationship between a behavior and its consequences. Contingency management, therefore, means changing behavior through the regulation of this relationship" (Berman, 1971). Contingency management has come to stand for the applied use of reinforcement theory.

Contingency Managed Education (CME) techniques have been applied to teaching in psychology (Malott and Svinicki, 1969; Ferster, 1968; Lloyd and Knutzen, 1969; McMichael and Corey, 1969). CME involves the use of educational objectives. Frequent sampling of students' repertoires combined with extensive remediation are key features. These objectives, like the teaching machines, increase the student's contact with the study materials, thereby increasing studying. The objectives can also direct the student to key points while requiring reading of the entire material. General objectives add relevance to specific instructional objectives (Clark, 1972).

The advantages of CME lie primarily in the demonstrated effectiveness of this approach to education. Another advantage is the inexpensiveness of its implementation. The cost of writing and copying objectives is by far less than the expense involved in the construction of a teaching machine or the use of additional textbooks as a learning supplement. Objectives can be easily revised as a result of performance analysis. However, as Michael (1969) pointed out, even excellent materials have little effect on the student if used improperly.

A disadvantage of CME is that its implementation in a large class requires the use of multiple choice questions for grading purposes. Another disadvantage inherent to CME is that students are forced to receive information in a "locked step" fashion which bores the fast learner and loses the slow learner.

Another approach to instruction is the Personalized System of Instruction (PSI), which was designed to facilitate learning and to avoid some of the problems inherent in the traditional lecture-readings format when applied to large college classes (Hess, 1974). PSI was initially referred to as the Keller Plan (Keller, 1968). Keller used the Skinner learning model and the techniques of operant conditioning to formulate what is now referred to as PSI. In the PSI approach, units of instruction are small, and are assisted by objectives and/or study guides designed to call attention to important points. PSI requires mastery of all materials presented, and has been shown to be an effective instructional method in 16 studies (Hursh, 1976). An advantage of PSI is that it allows students to proceed at their individual rates. However, this individualized pace can also be a disadvantage as some students may not finish the entire sequence of materials. Another advantage of PSI is its inexpensive implementation, a characteristic of CME as well. Multiple choice questions are widely used in PSI in order to facilitate immediate consequences in large classes.

Programmed Instruction (PI) is still another educational approach. PI has been defined as "a planned sequence of experiences, leading to proficiency in terms of stimulus-response relationships" (Espich and Williams, 1967). "Planned sequence" refers to the instructor's sequencing of the material to be learned. "In terms of stimulus-response relationships" refers to Skinner's definition of the components of learning, i.e., a stimulus in the environment, a response by the student, and consequences contingent upon the response. PI and Skinner's teaching machine follow Skinner's analysis of learning as closely as any educational technique described thus far, assuming that answering a frame correctly is actually reinforcing. PI forces the student to participate in the learning process by requiring him to respond to frequent questions. PI is primarily implemented with written textbooks where the student writes the answer in the textbook or in an accompanying workbook. PI allows the student to proceed at his individual rate without the traditional classroom distractions. Programmed materials are broken down into small steps which have been described as "optimally sized increments" (Pipe, 1966). Small steps for learning, combined with active participation on the part of the student and immediate knowledge of results, make PI one of the most potent forms of instruction. One drawback of PI is that there are no contingencies which assure that the student responds to each frame. The answers are usually readily available, and thus some students may not follow the sequencing determined by the instructor. The major drawback of the implementation of PI has been the great cost of developing an effective program. Programming skills require a great deal of time to develop, and PI is only as effective as the writing technique of the instructor (Espich and Williams, 1967).

Computer Assisted Instruction (CAI) is the most recently developed educational technology. CAI is, in essence, programmed instruction written by a person and taught by a computer. CAI consists of interactions between the machine and the learner, with the computer functioning as the instructor (Morgan, 1969). In CAI, programmed text is entered into the computer by the instructor. Students then interact with the instructional material via some input-output device, such as a teletype, Decwriter, or video screen. In this interaction, the computer presents text material and prints a question covering that text. The student then types an answer to the question into the computer. The computer then compares the student's answer with those already programmed by the instructor. The sequence of learning is determined by the instructor when the lesson is first written, and is followed by the computer. Both the training material and the instructional logic are stored in the computer.

The two main formats of CAI are drill and practice, and tutorial CAI. Both formats are response contingent, but differ in terms of the amount of computer involvement in instructional decision making. Drill and practice

CAI is the most basic form, and was the first developed. It was developed in 1964 in California elementary schools by Suppes (Suppes, Jerman and Brian, 1968). Drill and practice is the simplest with respect to computer involvement, and consists of material presented in a linear format. A linear format is one in which the student's performance has no effect upon instructional sequencing. Drill and practice is the most widely used form of CAI, and is an effective instructional method (Suppes, 1969; Suppes, Jerman and Groen, 1965; Fishman, Keller and Atkinson, 1969). The other level of CAI is tutorial or dialogue CAI. Tutorial CAI allows the student to construct natural language responses, and thus exercise control over the learning sequence via responses. Tutorial programs have the capabilities of making instructional branching contingent upon a single response or upon a subset of the student's response history (Atkinson, 1967).

There are many advantages of CAI. It has all of the positive features of PI, including individualized rate of response. CAI also provides immediate feedback to the student's responses. CAI allows for sequencing of instructional materials by the instructor. Unlike PI, CAI requires the student to make a response in order for the lesson to continue. CAI also serves as a pretest for the student to assure that the student is ready to proceed with the sequence of learning. CAI, as noted, allows branching to occur. Branching refers to selection of material which is determined by the student's performance. CAI also provides automatic scoring, which is a major component of the approach. Scoring greatly facilitates revision of programmed instruction writing style. This points to another advantage of CAI as compared with PI. This advantage is the ease with which lessons written on a computer can be modified. To alter programmed texts requires revision of an entire textbook. Multiple choice questions and questions requiring original answers are both possible using CAI. Cheating can be controlled for in CAI, because, unlike PI, the answers are not readily available to the student. Most subjective reports on CAI indicate that students find interaction with the computer to be reinforcing.

The disadvantages of CAI center around the availability of a computer to implement the program. The cost of a computer is much greater than the cost of mimeographing objectives, as in CME. Also, as with PI, CAI is very costly in time to implement. Writing effective CAI requires many revisions of style. One other drawback of CAI is that most instructors possess typing skills, but may not possess the skills required to enter text into a computer.

CAI also contains some elements of Computer Managed Instruction (CMI). CMI uses the computer to help the instructor administer and guide the instructional process, but relies upon separate hardware and learning materials. CMI has been shown to be the most cost effective method of sequencing instructional materials when dealing with large classes (Hansen, 1970; Laham, 1973; Fromer, 1972). The scoring and branching features of CAI incorporate the basic logic of CMI.

There are a large number number of CAI programs in use today. Three of the most successful and widely known are the Stanford Project, the PLATO System, and the GNOSIS System. The Stanford Project was developed at the Institute for Mathematical Studies in the Social Sciences Department at Stanford University. The initial CAI program used a PDP-1 computer. This has now been replaced by an IBM 1500 System which consists of an IBM 1800 Central Processing System with bulk storage maintained on tape and disk, a station controller, and other peripheral devices. PLATO, (Programmed Logic for Automated Teaching Operations), was developed at the University of Illinois. Instruction in over 30 subjects is available in PLATO, including a visual display which can simulate operant experiments. PLATO utilizes a large central computer. GNOSIS, which is available on the PDP-10 computer, incorporates the features of tutorial CAI.

Recent hardware developments with remote input-output terminals make CAI more feasible than ever before with respect to cost (Knezevich and Eye, 1970). The cost of a computer system has been declining rapidly since 1970. As shown by Knezevich and Eve (1970), with the rising costs of professors' salaries and the declining costs of computers, CAI is an even more cost effective method of instruction. CAI is most cost effective with small classes and self-instruction (Suppes, 1977). The special languages required to communicate with the computer have become very user oriented. A computer programmer is no longer required to create CAI lessons. A computer naive user can be guided by the CAI system to mastery of a language (Gerard, 1969).

In spite of the great proliferation of CAI languages available today, no CAI language is widely available for the PDP-8 mini-computer. The PDP-8 is widely used in psychological laboratories, with over 100 users in the SKED Users Group alone. CAI would be useful to document laboratory procedures, and to serve as a pretest to student utilization of the computer for research purposes. Learning computer skills on a computer provides a desensitization which can be beneficial in acquiring computer usage skills. CAI for the PDP-8 computer will be useful only if a CAI language is developed which is compatible with the PDP-8, and if that CAI language can be easily implemented by non-sophisticated users.

The purpose of the present study is to develop a CAI language for the PDP-8 computer which can be easily learned. This study will develop such a language, and will document it using lessons written in the language. Subjects will then be exposed to the lessons and their associated contingencies. The CAI language will be considered effective if the subjects are able to write instructional sequences on their own after exposure to the lesson.

METHOD

Subjects

Twelve students served as subjects. The subjects' computer experience varied from naive to experienced, with the majority of subjects being naive.

Apparatus

All programming was done on a PDP-8e computer, manufactured by Digital Equipment Corporation (DEC).[1] Input-output was handled by a Decwriter, also manufactured by DEC. An RK8 disc served as the OS/8 device. The software used to create the CAI program and the lessons consisted of U/W FOCAL[2] and DEC's Symbolic Editor.

The following equipment is necessary to use TEACH:

1. A PDP-8 computer with 8k or more of memory and an OS/8 device.
2. A U/W FOCAL source.
3. An input-output device.
4. A copy of TEACH.

Procedure

The language chosen to create a CAI language for PDP-8 computers was U/W FOCAL. The reasons behind this choice were several. First of

---

[1]DEC is the trademark of Digital Equipment Corporation of Maynard, Massachusetts.

[2]U/W FOCAL can be obtained from Jim Van Zee, Department of Chemistry, BG-10, University of Washington, Seattle, Washington.

all, U/W FOCAL contains the necessary features. Secondly, it is easily obtainable. Finally, programs developed in U/W FOCAL can easily be converted into other languages including either FORTRAN IV or BASIC.

The first step in writing a computer program is to develop a flowchart. Such a flowchart traces all of the basic components necessary for the program, without reference to a particular language format. The flowchart in Figure 1 is that from which TEACH was developed. It is a flowchart which can be adapted to any language. It is an outline of how an instructional lesson would operate. A lesson in CAI is a series of sequentially numbered frames. Each frame consists of text, a question concerning the text, and a series of right and wrong possible answers.

As shown in Figure 1, the program begins by initializing all variables, i.e., setting them equal to zero. The program asks the subject's name, and allows this information to be typed in. The subject's name and the date are stored in an output file called SCORE.FD. Next, the available lesson options are printed out, and the subject is allowed to choose one. The subject's choice is recorded in SCORE.FD, and the requested lesson file is opened for input. The first character is sampled by the program to determine whether or not it is a %. The first character in a CAI lesson written in TEACH must be a %. If this is not the case, the program prints teacher error, the subject's score file is closed, and the lesson ends. If the first character is a %, the program then selects the next character. This character is checked against other characters. If the character is a T, the program inputs and prints characters until a ? is found. The program then searches for a % and checks the next character. If that character is an R, the subject is allowed to enter his answer to the question which terminated the input of text. The subject's answer is terminated by a carriage return.

Next, the first programmed right answer is read in and compared with the subject's answer. If the two are the same, the number of the answer given by the subject is recorded in SCORE.FD, a congratulations message is printed to provide the subject with immediate feedback, and counter C is set equal to zero. The program then goes on to the next frame.

If, on the other hand, the subject's answer does not match the right answer with which it was compared, the next character is input. That character is checked by the program to determine if it is another possible right (R) answer. If a right answer is found, the program inputs the possible answer and compares it with the subject's answer. This continues until no more right answers remain, or until a correct match occurs.

If no possible right answers are found, the program then checks for possible wrong (W) answers. Wrong possible answers are then compared, one at a time, with the subject's answer. This can be terminated by a match of the subject's answer with one of the possible wrong answers. In this case, the program then searches for the corresponding %&. The %& allows a remedial message to be printed to the subject, clarifying the error. The number of the wrong answer is then recorded in SCORE.FD, and counter C is incremented by one. If C equals three, the subject has missed three consecutive questions, since every correct answer sets C back to zero. If C equals three, the subject is told that too many errors have been made, and the lesson ends. After a matched wrong answer, the program then goes on to the next frame.

If the subject gives an answer which does not match any of the possible answers, the program searches for a %S, which then initiates branching. Branching refers to a portion of text which may or may not be presented in a lesson depending upon the occurrence of another event. That event is the %S. After a %S, additional frames will be presented until either a matched answer or no additional %S options are found.

Two other characters should also be defined. The %J signifies the end of a lesson, and is placed at the end of a lesson. When encountered, the %J ends the lesson, prints out the subject's score, closes the output file, and saves this file in the computer for future reference by the instructor. The other character which is used in TEACH but has not yet been discussed, is the %E. This character is used when writing lessons. The %E can be placed at the end of a frame so that each frame can be tested individually prior to testing by the subjects. The %E is also a somewhat standard character in CAI, and is used to detect errors in the instructor's writing of the program.

The FOCAL program TEACH was developed from the flowchart in Figure 1. Figure 2 is an actual copy of TEACH in U/W FOCAL. The commands used in U/W FOCAL are summarized in Table 1. For additional information on U/W FOCAL, consult the U/W FOCAL manual.[1]

Next, CAI lessons documenting TEACH were written using the Symbolic Editor program to enter text into the computer. Three lessons were written, and are shown in Appendix A. The lessons were then tested by subjects.

---

[1]ibid.

Testing consisted of subjects running lessons one and two in Dr. Arthur Snapper's laboratory. The computer was started for the subjects, and TEACH was called into core. From this point, the subjects were not disturbed until the lessons were completed. The subjects were allowed to work at their own pace. The programmed writing style used in the lessons was revised several times based upon analysis of the subjects' results. Following these revisions, a criterion was established which required the subject to complete the lessons and to write a CAI lesson based on knowledge gained in lessons one and two. This was felt to be a satisfactory demonstration of the program's effectiveness as an instructional tool, and the ease with which the CAI program TEACH can be learned by those lacking an extensive computer background.

RESULTS

The program TEACH was first written with each frame of a lesson as an individual file on the computer. This permitted complex branching. However, it quickly filled up the available directory space on the computer, and thus was not a feasible design for a PDP-8 CAI language.

TEACH was then changed to its present format, in which each entire lesson constitutes an individual file on the computer. TEACH allows remedial messages following matched wrong answers and limited branching in the present format, as well as a detailed scoring system. Two lessons were then written to document and test TEACH. It should be noted that a third lesson describing the Symbolic Editor was made available, and was adapted from a tested and revised lesson available in GNOSIS on the PDP-10 computer. Lesson four, which is shown in Figure 3, was written as a short model lesson to assist subjects in writing their own CAI lessons.

Lessons one and two were each tested by 12 subjects. Testing was conducted in such a manner that after an error pattern was observed in subjects' scores, testing was halted and the error was revised, based upon the data received from the subjects. The criterion for revisions was somewhat subjective, but generally followed these three rules:

1. If subjects consistently missed a particular frame, the frame was rewritten.
2. If subjects consistently gave the same wrong answer to a particular frame, the frame was searched for the misleading section and revised.
3. If the lessons were completed by the subjects with no consistent errors, the effectiveness of these lessons was tested by requiring the subjects to write a short CAI lesson.

Six subjects tested the lessons in their early format, and the data from these tests led to revisions of lessons one and two. Also as a result of the data collected from these subjects, additional right and wrong answers were added which had not been initially anticipated. %S branching options were also added to frames to which wrong answers were consistently given, and to frames in which the content was particularly important.

The next six subjects completed the revised lessons, and no error patterns were detected. However, passages verbally reported as confusing were revised, and additional unanticipated answers were added based upon the subjects' results. Of the final six subjects to test lessons one and two, four were able to write instructional sequences of their own. One subject dropped out of the study due to time limitations. The final subject required extensive work on the Symbolic Editor, a delay which made it impossible to include that subject's results in the study, due to the termination of the study.

DISCUSSION

The results of the present study indicate that TEACH, a CAI language developed for the PDP-8 computer, is easily learned and can be effectively used as an instructional technique.

CAI fits Skinner's stimulus-response-consequence learning theory best of all the educational methods discussed. In CAI, the stimulus is a discrete frame to which the student must respond in order for the lesson to continue. All responses have immediate consequences contingent upon the correctness of the response. CAI was used in Psychology 535, Digital Control, and the students were required to complete unit feedback forms. Of those completed, approximately 80 per cent were highly positive with respect to the use of CAI.

TEACH can be incorporated into a useful contingency managed approach to the acquisition of laboratory skills. To be allowed to conduct an experiment involving computers, the student would first be required to complete CAI modules on the computer which would teach the required computer skills. Such a method of training would be much more efficient and standardized than the current method of verbal instruction. Supplemental materials could also be coordinated with CAI when such materials were available. This was done in Psychology 535 where CAI modules written in GNOSIS to teach the Symbolic Editor were supplemented by DEC's <u>Introduction to Programming</u>.[1]

[1]Small Systems Software Documentation Group Digital Equipment Corporation, <u>Introduction to Programming</u>. PDP-8 Handbook Series, Digital Equipment Corporation, Maynard, Massachusetts, 1975.

Another possible use of TEACH would be the establishment of a departmental CAI library. This library would be composed of short lessons written by students covering various skill areas, such as logarithms, matching law, and other important areas not always familiar to all students. The students writing the various CAI lessons would derive additional personal benefit from such a library in the additional knowledge gained from actually teaching the lessons.

One of the major improvements made in TEACH over other CAI languages, such as GNOSIS, is the scoring system. Data is the best method of effectively revising programmed instruction writing style. With GNOSIS, the instructor must search through the student's entire print-out, which is often very lengthy, to find the errors made so that poor frames can be revised. With TEACH, all answers are numbered, and an accurate listing of the student's answer to each question is presented to the student, and is also saved on the computer for the instructor's evaluation.

There are several constraints which must be followed when writing lessons using TEACH. Due to the fact that a text frame is terminated by a question mark (?), this symbol cannot appear in the text except as a terminator. This is shown in Frame 1 of Figure 3. The question mark (?) appears in the only possible place. The per cent (%), as mentioned, is a major control character, and should only be used as shown in Figure 3. Finally, a %J must be the last line of a lesson in order for the lesson to stop after the final frame.

There are several minor drawbacks of TEACH. Branching is very limited, and no repeat option is available. Repeat in GNOSIS allows a frame to be repeated any number of times until it is answered correctly. This can be done in TEACH, but would involve using the %S option over and over, retyping the entire frame each time. In addition, GNOSIS can scan for extra characters in answers. This means that if the correct answer to a question was the single word "relay", and the answer typed was "the relay card", GNOSIS calls the answer correct and prints a message that the extra words were "the card". This can also be done to a limited extent using TEACH. TEACH allows the instructor to list potential wrong answers so that if there was a high probability that an answer would contain an extra word, that answer could be scanned for, and, if found, a remedial message, coordinated with the particular wrong answer, could be given to the student. This would involve use of the %& option. Figure 3 demonstrates the use of %&.

The primary goal of this study was the development of a CAI language for the PDP-8 computer. As with all computer programs, revisions and improvements can always be made.

TEACH was written in U/W FOCAL because U/W FOCAL is the easiest and most powerful language for a beginning programmer. In addition, U/W FOCAL is easily transferable to BASIC, FORTRAN IV, and Machine Language. BASIC would be a good language in which to write TEACH because it is available on micro-computers. However, there are many different versions of BASIC for micro-computers, a fact which would limit the program's usefulness. FORTRAN IV is widely transferable and could be used to write a CAI language. However, the optimal language into which to convert TEACH would be Machine Language. All languages, such as BASIC, FORTRAN IV, and FOCAL, are originally programmed in Machine Language. Thus, writing TEACH in Machine Language would eliminate the need for FOCAL, and would speed up the operation of the program.

CAI has the potential to use what is currently known about behavior analysis to generate an instructional sequence. Through testing and improvements, CAI can be the best educational technique available as the costs of computers decrease and the materials available for CAI increase.

Table 1: U/W FOCAL commands.

TABLE 1

U/W FOCAL COMMANDS

Basic Commands

Command	Abbreviation	Function
ASK	A	Allows a variable to be set equal to a number from the keyboard.
COMMENT	C	Anything typed following a "C-" is not read by the computer and has no effect on the program.
IF	I	Allows the program to branch depending on the occurrence of a character.
TYPE	T	Prints (outputs) text.
ZERO	Z	Used at the start of a program to set all variables equal to zero.

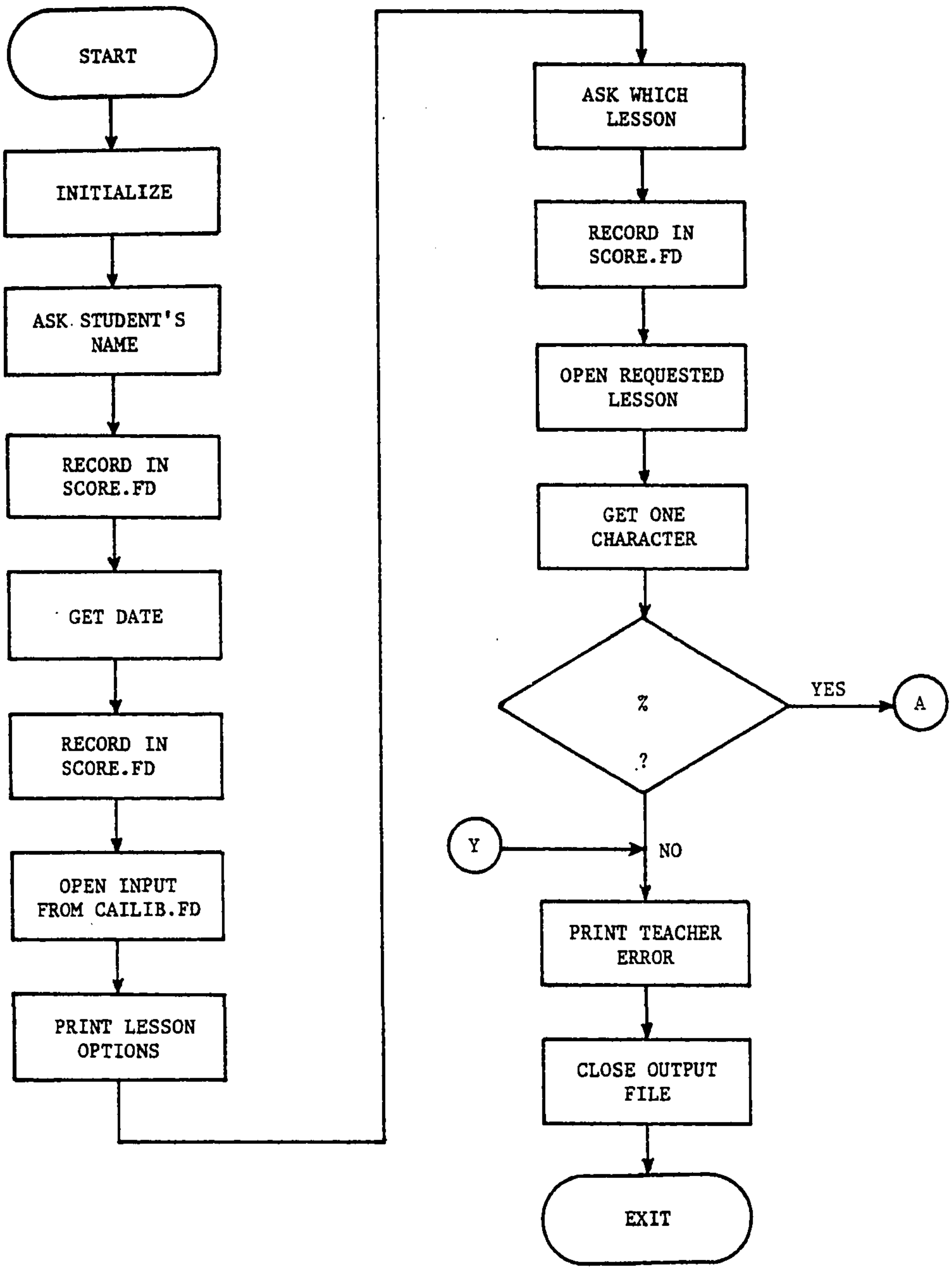
Function Commands

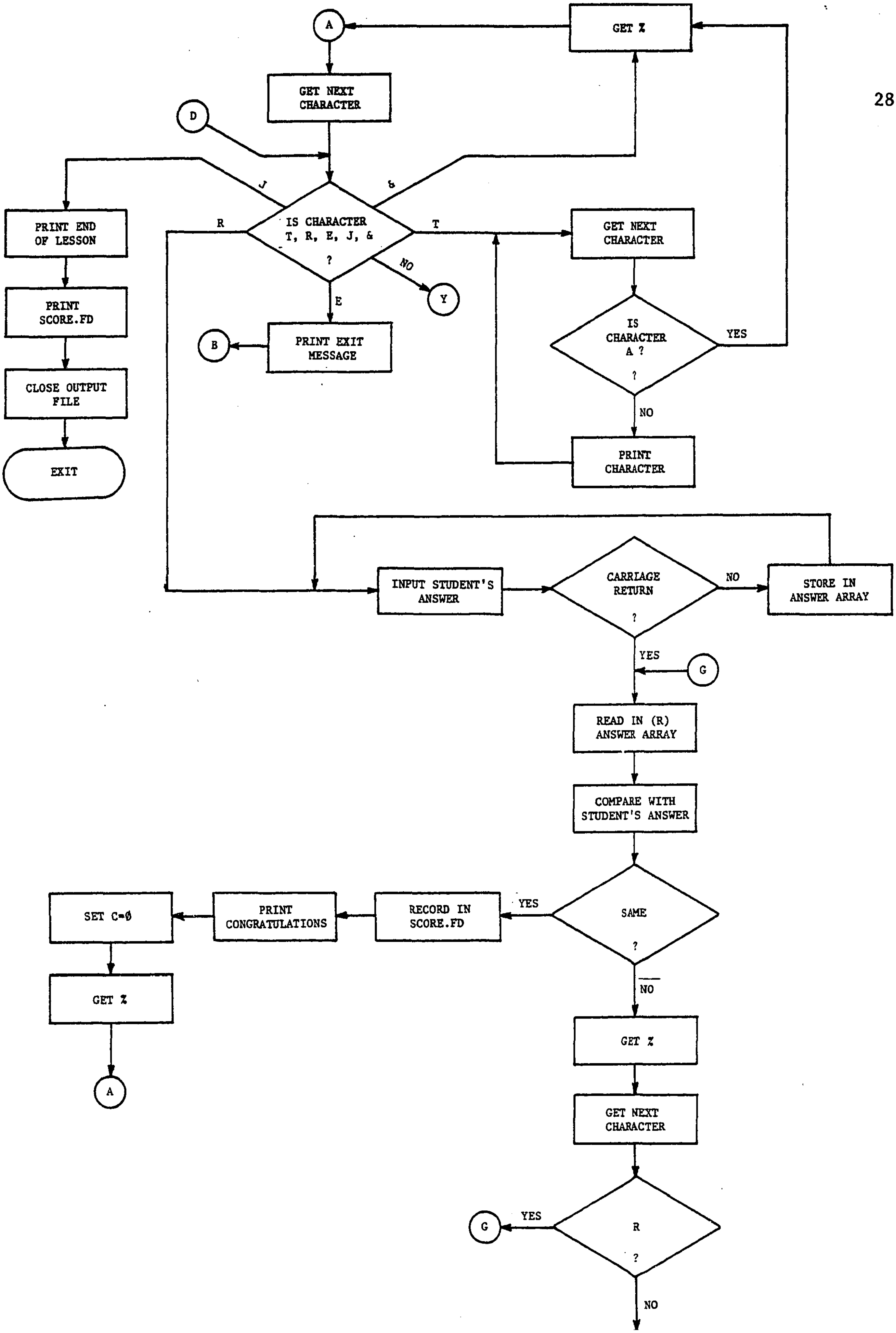
Command	Abbreviation	Function
	FIN()	Inputs a single character from a lesson file.
	FIND()	Inputs characters from a file up to a specified character.
	FOUT	Outputs characters.

File Commands

Command	Abbreviation	Function
OPEN INPUT	O I	Opens input from either a separate file, such as a lesson, or from the TTY (keyboard).
OPEN RESTORE INPUT	O R I	Opens input from a previously opened file; input begins at point where input last left off.
ECHO	E	Allows text to be printed.
OPEN OUTPUT	O O	Opens a new output file.
OPEN RESTORE OUTPUT	O R O	Opens output to a previously opened output file.
OUTPUT CLOSE	O C	Closes an output file.

Figure 1: Flowchart from which TEACH was developed.


START
INITIALIZE
ASK STUDENT'S NAME
RECORD IN SCORE.FD
GET DATE
RECORD IN SCORE.FD
OPEN INPUT FROM CAILIB.FD
PRINT LESSON OPTIONS
ASK WHICH LESSON
RECORD IN SCORE.FD
OPEN REQUESTED LESSON
GET ONE CHARACTER
%
?
YES
A
Y
NO
PRINT TEACHER ERROR
CLOSE OUTPUT FILE
EXIT


A
GET %
GET NEXT CHARACTER
D
J
&
PRINT END OF LESSON
R
IS CHARACTER T, R, E, J, & ?
T
GET NEXT CHARACTER
NO
Y
E
PRINT SCORE.FD
B
PRINT EXIT MESSAGE
IS CHARACTER A ?
YES
CLOSE OUTPUT FILE
NO
EXIT
PRINT CHARACTER
INPUT STUDENT'S ANSWER
CARRIAGE RETURN ?
NO
STORE IN ANSWER ARRAY
YES
G
READ IN (R) ANSWER ARRAY
COMPARE WITH STUDENT'S ANSWER
SET C=Ø
PRINT CONGRATULATIONS
RECORD IN SCORE.FD
YES
SAME ?
NO
GET %
GET %
A
GET NEXT CHARACTER
G
YES
R ?
NO

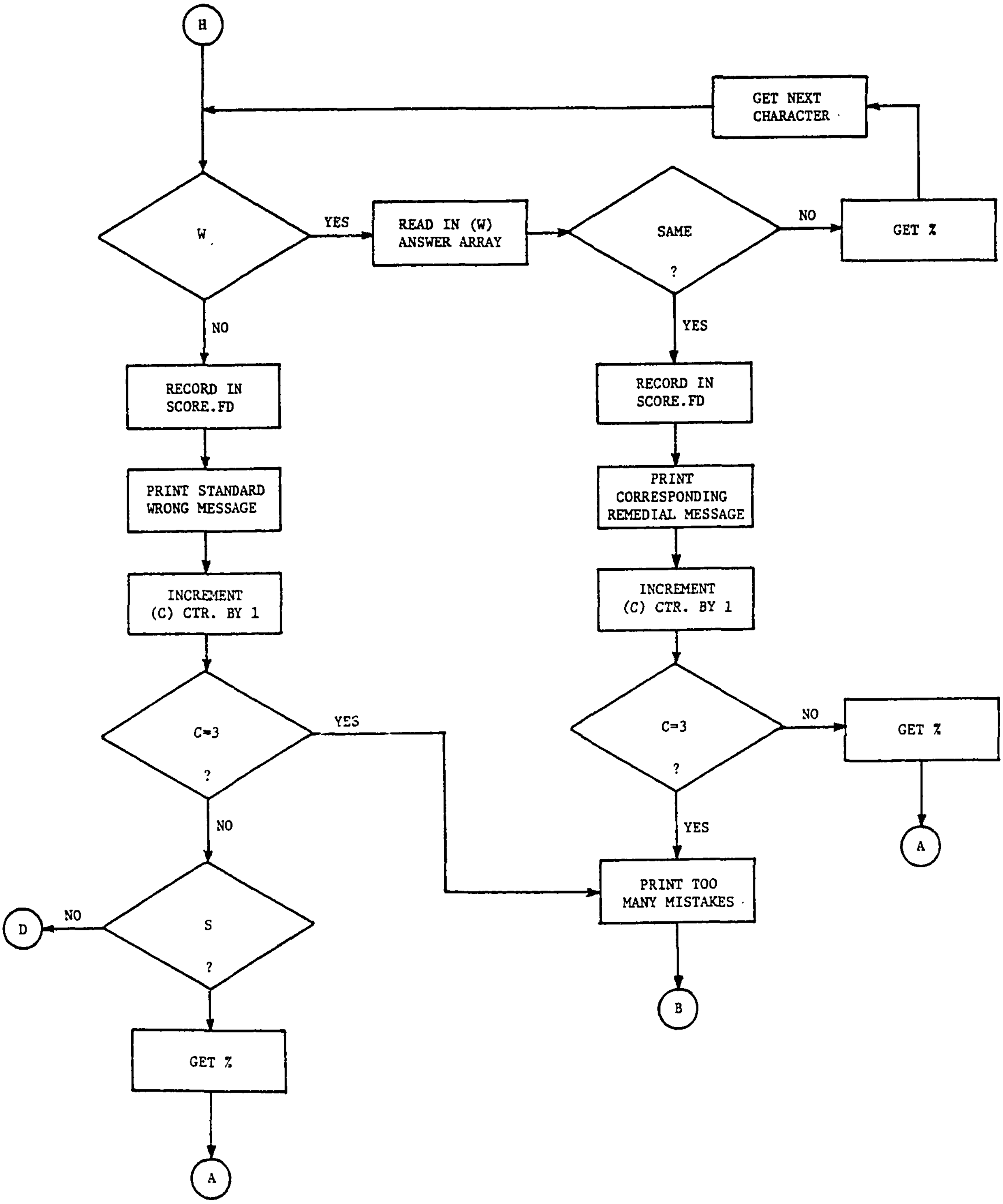

H
GET NEXT CHARACTER
W
YES
READ IN (W) ANSWER ARRAY
SAME ?
NO
GET %
NO
YES
RECORD IN SCORE.FD
PRINT STANDARD WRONG MESSAGE
INCREMENT (C) CTR. BY 1
C=3 ?
YES
NO
S ?
NO
D
GET %
A
RECORD IN SCORE.FD
PRINT CORRESPONDING REMEDIAL MESSAGE
INCREMENT (C) CTR. BY 1
C=3 ?
NO
GET %
A
YES
PRINT TOO MANY MISTAKES
B

Figure 2: The U/W FOCAL program TEACH.

```
C U/W-FOCAL:  TEACH   07/15/77

01.01 Z;C-ZEROS ALL VARIABLES
01.02 C-VARIABLES USED: M,T,W,J,E,&,%,C,R,K,A,B,N
01.03 C-M IS THE LESSON NUMBER
01.04 C-R&W=RIGHT AND WRONG ANSWER POSSIBILITIES
01.05 C-J SIGNIFIES THE END OF THE LESSON
01.06 C-& ALLOWS YOU TO PRINT REMEDIAL TEXT AFTER A WRONG
01.07 C-C COUNTS NUMBER OF MISSES
01.08 C-A STORES STUDENT ANSWER
01.09 C-B STORES PROGRAMMED ANSWER
01.10 T 'TYPE YOUR NAME, FOLLOWED BY A CARRIAGE RETURN',!
01.20 O O SCORE,E;X FOUT(FIND(141));O D;T !;O O TTY:
01.21 T !,!
01.22 T 'HERE ARE YOUR OPTIONS!!!',!
01.30 O I CAILIB,E;X FIND(154);O I TTY:,E;C- PRINTS LESSON OPTIONS
01.40 A 'WHICH LESSON DO YOU WANT?(ANSWER MUST BE A DIGIT) 'M
01.50 O R O;T %3;T 'LESSON ',M;T !,!;O O TTY:
01.55 T !;T '                        LESSON:'M;T !,!,!
01.60 O I LESS(M) 1.90;S X=FIN();I (X-165)3.1,2.1,3.1;C-CHECK FOR %
01.90 O I TTY:,E;T 'LESSON NOT FOUND-START OVER! ';O C;GO;ALLOWS FOR ERROR

02.05 X FIND(165);C-GETS NEXT %
02.10 S X=FIN();C- GETS NEXT CHAR.
02.20 I (X-212) 2.3,5.1,2.3;C-CHECK FOR T
02.30 I (X-202) 2.4,6.1,2.4;C-CHECK FOR J
02.40 I (X-166) 2.5,2.05,2.5;C-CHECKS FOR REMED. LOOP
02.50 I (X-197) 3.1,7.1,3.1;C-CHECKS FOR E

03.01 C-THIS IS A DEFAULT LOOP FOR WHEN NO LEGIT. CHAR. FOLLOWS A %
03.10 O I TTY:,E;T 'TEACHER ERROR LESSON:',M;T ' ENDING';O C;L E

04.01 C-STUDENTS ANSWER IN INPUT
04.10 F K=1,1,60;S A(K)=0
04.13 O I TTY:,E;C-ANSWER TYPED IN
04.15 S K=1
04.20 S A(K)=FIN()
04.22 I (A(K)-255) 4.25,4.80,4.25;C-RUBOUT
04.25 I (A(K)-141) 4.30,4.90,4.30;C-CR TERM. ANSWER
04.30 S K=K+1
04.40 GOTO 4.20
04.80 T '\';C-ECHOS RUBOUT
04.81 S K=K-1
04.82 G 4.20
04.90 R

05.01 C-TEXT PRINTING LOOP
05.10 A T;T 'FRAME:'T;T !,!;O R I,E;X FOUT(FIND(191));T !,!;O R I;D 4
05.15 O R I
05.16 X FIND(165)
05.18 S X=FIN()
05.20 I (X-210) 14.10,10.05,14.10
05.25 F K=1,1,N;D 13
05.30 C-RETURNS HERE IF R MATCHES
05.35 O R O;T 'RIGHT ',R;T !;O O TTY:
05.40 S C=0
05.45 F K=1,1,60;S B(K)=0
05.50 S B=0
05.60 T 'GREAT YOU ARE RIGHT!!',!,!
05.65 O R I
05.70 X FIND(165)
05.75 S X=FIN()
05.80 I (X-212) 5.85,5.01,5.85;C- CHECKS FOR T
05.85 I (X-202) 5.70,6.10,5.70;C-CHECKS FOR J-OR ELSE WAITS FOR T

06.01 C- THIS IS THE END OF LESSON LOOP-CODED WITH A 'J'
06.10 O I TTY:,E;T 'END OF LESSON:',M;O C;T !
06.20 O I SCORE,E;X FIND(154);O C;L E;C-PRINTS SCORE OF STUDENT

07.01 C- THIS IS THE ERROR LOOP-COMES HERE ON AN 'E'
07.10 O I TTY:,E;T 'EXITING FROM LESSON';O C;L E

08.01 C- COMES HERE IF NO R OR W MATCHS, AFTER PRINTING WRONG AND
08.02 C- RECORDING IN SCORE.FD
08.06 C-(S) IS A SPEC. CHAR. THAT ALLOWS BRANCHING
08.10 O R I,E; X FOUT(FIND(191));T !;C- INPUTS TEXT AND ASKS QUESTION
08.20 O R I;D 4
08.30 GOTO 5.15
```

```
10.01 C-PROGRAMMED ANSWER READ IN AND STORED
10.05 A R;C-SETS R FOR RECORDING
10.10 O R I
10.15 S N=1
10.20 S B(N)=FIN()
10.25 I (B(N)-141) ,5.25;C-CHECKS FOR CR WHICH TERM. ANSWER
10.30 Y N
10.35 GOTO 10.20

13.01 C-COMPARISON ROUTINE FOR (R);COMES HERE FROM 5.25
13.10 I (A(K)-B(K)) 13.50,13.20,13.50
13.20 R;C-GOES TO 5.30 IF A R MATCH
13.50 BREAK 13.60
13.60 O R I;X FIND(165)
13.70 GOTO 5.18;C-TO CHECK FOR NEXT R

14.01 C-COMES HERE IF NO MORE (R) ANSWERS
14.10 A W;C-GETS NUMBER
14.23 S N=1;C-SETS VAR. N EQUAL TO 1
14.25 S B(N)=FIN()
14.30 I (B(N)-141) ,17.10;C-AFTER CR GO TO 17
14.35 Y N
14.40 GOTO 14.25

15.01 C-STANDARD WRONG OPTION(IF NO MORE (W) ANSWERS)
15.10 O R O;T 'WRONG ',T;T !;O O TTY:;C-RECORDS IN SCORE.FD
15.13 BREAK 15.15
15.15 O I TTY:,E;T 'WRONG',!,!
15.20 Y C;I (C-3) 15.30,16.10,15.30;C-IF 3 WRONG ANS. IN ROW, LESS. ENDS
15.30 O R I;I (X-211) 2.20,8.10,2.20;C- CHECKS FOR %S FOR BRANCHING

16.01 C-IF TOO MANY ERRORS IN A ROW
16.10 O I TTY:,E;T 'TOO MANY ERRORS',!
16.20 T 'YOU HAVE MADE 3 ERRORS IN A ROW!!!!',!
16.25 T 'YOU WILL HAVE TO START LESSON',M;T ' OVER!',!
16.30 GOTO 7.10

17.01 C-LOOP FOR (W) MATCH RETURN AND REM. MESS.& CHECKS (C)
17.10 F K=1,1,N;DO 18
17.11 O R C;T 'WRONG ',W;T !;O O TTY:;C-RETURNS HERE ON (W) MATCH
17.12 F K=1,1,60;S B(K)=0
17.20 S B=0
17.21 O I TTY:,E;T 'WRONG!!',!;C-PRINTS WRONG
17.30 O R I;X FIND(165);C-GETS %
17.35 S X=FIN();I (X-166) 17.40,17.60,17.40;C-REMEDIAL MESSAGE
17.40 I (X-212) 17.45,5.01,17.45;C-CHECKS FOR T
17.45 I (X-202) 17.50,6.10,17.50;C-CHECKS FOR J
17.50 I (X-197) 17.30,7.10,17.30;C-CHECKS FOR E
17.60 O R I,E;X FIND(165);T !;C-INPUTS %S TEXT AND HALTS AT ?
17.65 Y C;I (C-3) 17.70,16.10,17.70;C-INC. C AND CHECK C
17.70 O R I;S X=FIN()
17.75 I (X-212) 17.80,5.01,17.80;C-CHECKS FOR T
17.80 I (X-202) 17.85,6.10,17.85;C-CHECKS FOR J
17.85 X FIND(165);GOTO 17.70

18.01 C-COMPARISON ROUTINE FOR (W)
18.10 I (A(K)-B(K)) 18.50,18.20,18.50
18.20 R;C-RETURNS TO 17.11 IF (W) MATCH
18.50 BREAK 18.60;C-COMES HERE IF (W) DOES NOT MATCH
18.60 O R I;X FIND(165);C-GETS %
18.65 O R I;X FIND(165);S X=FIN();I (X-215) 15.10,14.20,15.10
```

Figure 3: A sample CAI lesson written in TEACH.

```
%T01 THIS IS A SAMPLE LESSON TO ILLUSTRATE HOW TO WRITE A
CAI LESSON.  THE LESSONS THAT YOU WRITE SHOULD FOLLOW
THIS FORM.  ALL YOU DO IS CREATE LESS#.FD AND THEN BEGIN TYPING.
EVERYTHING IN THIS LESSON WAS TYPED BY ME. YOU SHOULD
ALWAYS RUN YOUR LESSONS FIRST TO CHECK FOR ERRORS.
I NUMBER QUESTIONS TO CORRESPOND WITH THE TEXT THAT
THEY COVER.  QUESTIONS MUST BE TERMINATED BY A QUESTION MARK.
Q1- WHAT TERMINATES A QUESTION?
%R11 ?
%R12 QUESTION MARK
%R13 A QUESTION MARK
%R14 A ?
%W11 A CARRIAGE RETURN
%& NO, A QUESTION MARK.  THE ? TERMINATES BOTH THE QUESTION
AND THE TEXT.
%T02 THIS WOULD BE THE SECOND FRAME OF TEXT.
IN THIS FRAME I WILL DEMONSTRATE HOW TO USE THE PERCENT S
OPTION.
Q2- WHICH FRAME IS THIS?
%R21 2
%R22 TWO
%R23 THE SECOND
%R24 THE SECOND FRAME
%W21 4
%& NO, ITS THE SECOND FRAME OF LESSON 4.
%W22 FOUR
%& NO, ITS THE SECOND FRAME OF THE FOURTH LESSON.
%S IF THE STUDENTS ANSWER DID NOT MATCH ANY OF THE ANSWERS
THAT I HAD LISTED, THE PROGRAM WOULD CHECK FOR A
PERCENT S.  IF IT FINDS ONE THEN BRANCHING WILL OCCUR.
THAT MEANS THAT TEXT WILL BE PRINTED THAT WOULD NOT
HAVE BEEN PRINTED IF THE STUDENT HAD MATCHED ONE OF THE
POSSIBLE ANSWERS THAT I HAD LISTED.  THE PERCENT
S OPTION MUST BE TERMINATED BY A QUESTION AND A QUESTION MARK.
THEN YOU LIST POSSIBLE RIGHT AND WRONG ANSWERS JUST
LIKE YOU WOULD FOR A NORMAL FRAME.  I NUMBER THE
POSSIBLE ANSWERS WITH AN EXTRA DIGIT SO THAT I CAN
DIFFERENTIATE BETWEEN THEM FOR SCORING PURPOSES.
Q2- AFTER A PERCENT S CAN YOU HAVE MORE THAN ONE LINE OF TEXT?
%R211 YES
%W211 NO
%& OF COURSE YOU CAN, I JUST HAD OVER 10 LINES OF TEXT
FOLLOWING A PERCENT S.
%T03 THE LAST FRAME HAD ONLY ONE PERCENT S OPTION.
THUS, NO MATTER WHAT THE STUDENT ANSWERED THE PROGRAM CAME HERE.
THAT IS BECAUSE IF THE STUDENT MATCHED ONE OF THE LISTED ANSWERS,
THE PROGRAM AUTOMATICALLY GOES TO THE NEXT FRAME.  AND
IF THE STUDENT DID NOT MATCH ANY OF THE LISTED ANSWERS,
THE PROGRAM CHECKS FOR ANOTHER PERCENT S.  IF I
HAD FELT ADDITIONAL REMEDIATION MAY HAVE BEEN NECESSARY, I COULD
HAVE ADDED ANOTHER PERCENT S.
Q3- CAN YOU HAVE MORE THAN ONE PERCENT S OPTION PER FRAME?
%R31 YES
%W31 NO
%& OF COURSE YOU CAN.
%J
```

REFERENCES

Atkinson, R. C. Computerized instruction and the learning process. Technical Report No. 122, Psychology Series, Institute for Mathematical Studies in the Social Sciences, Stanford Univeristy, Stanford, California, September 15, 1967.

Berman, M. L. Introduction: Contingency management issue. Educational Technology, April, 1971, 11, 7-10.

Bunderson, C. V. cited in Holtzman, W. H. Computer-assisted instruction, testing, and guidance. New York: Harper & Row, 1970, Chapter 4.

Clark, D. C. Using instructional objectives in teaching. Glenview, Illinois: Scott-Foresman & Co., 1972.

Espich, J. E. and Williams, B. Developing programmed instructional materials. Belmont, California: Fearon Publishers, Inc., 1967.

Ferster, C. B. Individualized instruction in a large introductory course. The Psychological Record, 1968, 18, 521-532.

Fishman, E. J., Keller, L. and Atkinson, R. C. Massed versus distributed practice in computerized spelling drills. In Atkinson, R. C. and Wilson, H. A. (Eds.) Computer-assisted instruction: A book of readings. New York: Academic Press, 1969, 227-237.

Fromer, R. Distinctions between CAI and CMI. Educational Technology, May, 1972, 12, 30-31.

Gerard, R. W. Shaping the mind: Computers in education. In Atkinson, R. C. and Wilson, H. A. (Eds.) Computer-assisted instruction: A book of readings. New York: Academic Press, 1969, 15-39.

Hansen, D. N. The role of computers in education during the 70's. CAI Center Technical Memo, No. 15, Florida State University, Tallahasee, Florida, May 15, 1970, 20.

Hess, J. H. Keller plan instruction: Implementation problems. In Sherman, J. G. (Ed.) Personalized system of instruction: 41 Germinal papers. Menlo Park, California: W. A. Benjamin, 1974, 136-147.

Hursh, D. E. Personalized systems of instruction: What do the data indicate? Journal of Personalized Instruction, September, 1976, 2, 91-105.

Keller, F. S. Good-bye, teacher... Journal of Applied Behavior Analysis, 1968, 1, 79-89.

Knezevich, S. J. and Eye, G. G. (Eds.) Instructional technology and the school administrator. Washington, D.C.: The American Association of School Administrators, 1970.

Laham, S. L. The design and development of an educational data processing system. Masters thesis, Western Michigan University, 1973.

Lloyd, K. E. and Knutzen, J. J. A self-paced programmed undergraduate course in the experimental analysis of behavior. Journal of Applied Behavior Analysis, 1969, 2, 125-133.

Malott, R. W. and Svinicki, J. G. Contingency management in an introductory psychology course for one thousand students. The Psychological Record, 1969, 19, 545-556.

McMichael, J. S. and Corey, J. R. Contingency management in an introductory psychology course produces better learning. Journal of Applied Behavior Analysis, 1969, 2, 79-84.

Michael, J. L. Management of behavioral consequences in education. Revista Interamericana de Psicologia, 1969, 205-252.

Michael, J. L. The essential components of effective instruction and why most college teaching is not. In Keller, F. S. and Ribes-Inesta, E. (Eds.) Behavior modification-Applications to education. New York: Academic Press, 1974, 163-176.

Morgan, R. M. A review of educational applications of the computer, including those in instruction, administration and guidance: A series two paper from ERIC at Stanford. ERIC Clearinghouse on Educational Media and Technology, Stanford, California, August, 1969, 13.

Pipe, P. Practical programming. Huntington, New York: Robert E. Krieger Publishing Company, 1966.

Skinner, B. F. Cumulative record. New York: Appleton-Century-Crofts, Inc., 1959.

Suppes, P. Computer technology and the future of education. In Atkinson, R. C. and Wilson, H. A. (Eds.) Computer-assisted instruction: A book of readings. New York: Academic Press, 1969, 41-47.

Suppes, P. The role of psychological models in computer-assisted instruction. Behavior Research Methods & Instrumentation, April, 1977, 9, 66.

Suppes, P., Jerman, M. and Brian, D. Computer-assisted instruction: Stanford's 1965-66 arithmetic program. New York: Academic Press, 1968.

Suppes, P., Jerman, M. and Groen, G. Arithmetic drills and review on a computer-based teletype. Technical Report No. 83, Psychology Series,

Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, California, November 5, 1965.

APPENDIX A

THREE LESSONS WRITTEN IN TEACH

Lesson 1: How to use CAI
Lesson 2: How to write CAI lessons
Lesson 3: Basic Symbolic Editor commands

```
%T01 WELCOME TO LESSON 1 OF THE PDP-8 COMPUTER ASSISTED INSTRUCTION
SERIES.  TO INTERACT WITH THIS COMPUTER YOU WILL NEED TO
FOLLOW A FEW SIMPLE RULES.  THE FIRST THING THAT YOU MUST KNOW
IS HOW TO ANSWER QUESTIONS FROM THE COMPUTER.  AN ANSWER
IS TERMINATED BY A CARRIAGE RETURN.  THUS, ALL OF YOUR ANSWERS
MUST BE LESS THAN ONE LINE LONG AND MUST BE FOLLOWED BY A CARRIAGE
RETURN.
Q1- WHICH KEY TERMINATES YOUR ANSWERS TO THE COMPUTERS QUESTIONS?
%R11 CARRIAGE RETURN
%R12 THE CARRIAGE RETURN
%R13 THE CARRIAGE RETURN KEY
%R14 A CARRIAGE RETURN
%R15
%R16 C R
%R17 C.R.
%R18 CR
%W11 PERIOD
%& NO, YOU DO NOT NEED A PERIOD AFTER YOUR ANSWERS.  IN FACT
IF YOU ADD EXTRA CHARACTERS YOUR ANSWER WILL BE COUNTED WRONG!!
%S THE CORRECT ANSWER WAS A CARRIAGE RETURN.  SINCE THE CARRIAGE
RETURN TERMINATES ALL ANSWERS, THEN NATURALLY THE MAXIMUM LINE
LENGTH IS ONE LINE
WHAT TERMINATES AN ANSWER?
%R111 CARRIAGE RETURN
%R112 THE CARRIAGE RETURN
%R113 THE CARRIAGE RETURN KEY
%R114 A CARRIAGE RETURN
%R115
%R116 C R
%R117 C.R.
%R118 CR
%W111 ONE LINE
%& WELL, YOU WERE ON THE RIGHT TRACK.  ONE LINE IS THE MAXIMUM
LENGTH OF AN ANSWER.  THAT IS DUE TO THE FACT THAT A CARRIAGE RETURN
TERMINATES AN ANSWER AND A LINE.
%T02 THERE ARE A FEW KEYS THAT YOU NEED TO REFER TO BY TYPING THEIR
NAME, DUE TO THE FACT THAT THEY OPERATE DIRECTLY UPON
THE PROGRAM.  ONE OF THESE CHARACTERS IS THE RUBOUT.  IF
YOU NOTICE AN ERROR TYPING AND YOU NOTICE IT BEFORE
YOU HAVE TYPED ANOTHER CHARACTER, YOU CAN ERASE IT WITH THE
RUBOUT KEY.  BUT IF YOU HAVE TYPED ANOTHER CHARACTER AFTER THE
ERROR, YOU WILL JUST HAVE TO LIVE WITH YOUR ERROR AND LET THE
COMPUTER THINK THAT YOU ARE A DUMMY!!
Q2- WHICH KEY ERASES THE LAST CHARACTER TYPED(HINT:TYPE THE WORD
    BECAUSE IF YOU ACTUALLY TYPE THE CHARACTER IT WILL
    ERASE ITSELF)?
%R21 RUBOUT
%R22 THE RUBOUT
%R23 THE RUBOUT KEY
%R24 RUBOUT KEY
%W21 THE CARRIAGE RETURN
%& NO, THE CARRIAGE RETURN TERMINATES YOUR ANSWER.  THE RUBOUT
KEY CAN ERASE ERRORS IF THE ERROR IS THE LAST CHARACTER THAT
YOU TYPED.
%W22 THE CTRL C
%& NO, WE WILL DISCUSS CTRL C IN THE NEXT FRAME.  THE ANSWER
TO THE QUESTION WAS THE RUBOUT KEY.
%W23 CTRL C
```

```
%& NO, THE CTRL C WILL BE EXPLAINED IN THE NEXT FRAME.  THE RUBOUT
KEY CAN ERASE THE LAST CHARACTER THAT YOU TYPE.
%S IF YOU ANSWERED BY PRESSING THE RUBOUT KEY, YOU WERE ACTUALLY
CORRECT.  HOWEVER YOU DID NOT LISTEN TO MY HINT.  SINCE, THE
RUBOUT ERASES THE LAST CHARACHTER TYPED IF YOU PRESSED
THE RUBOUT KEY INSTEAD OF TYPING THE WORD RUBOUT THE
COMPUTER NEVER "SAW" YOUR ANSWER.  TYPE THE WORD
COMPUTER AND DELIBERATELY MAKE AN ERROR AND CORRECT IT?
%R211 COMPUTER
%W211 COMPUTER
%& YOU SHOULD NOT HAVE ADDED THAT FINAL SPACE
%S I DONT KNOW EXACTLY WHAT YOU DID, HOWEVER I DO KNOW THAT
YOU MADE AN ERROR.  TRY AGAIN TO TYPE COMPUTER AND USE THE
RUBOUT TO CORRECT A DELIBERATE ERROR?
%R221 COMPUTER
%W221 THE WORD COMPUTER
%& THE CORRECT ANSWER WAS JUST ONE WORD---  COMPUTER
%T03 THE NEXT THING THAT YOU SHOULD KEEP IN MIND IS THAT THE
CTRL C IS THE "EXIT" CHARACTER.  IF YOU WISH TO EXIT FROM THE
LESSON, WITHOUT FINISHING, YOU SHOULD PRESS CTRL C. ALSO CTRL C
IS TO BE USED IF THE LESSON SHOULD QUIT RUNNING.  TO TYPE A CTRL C
YOU SHOULD HOLD THE CTRL KEY DOWN AND PRESS THE C KEY.
Q3- TO EXIT FROM A LESSON WHAT KEY DO YOU PRESS WHILE HOLDING
THE CTRL KEY DOWN?
%R31 C
%R32 THE C
%R33 THE C KEY
%W31 THE CTRL KEY
%& NO, THE QUESTION ASKED WHICH KEY IN ADDITION TO THE CTRL
KEY---THATS THE C KEY!!
%W32 THE CTRL KEY AND THE C KEY
%& YOU ARE CORRECT IN THAT PRESSING THOSE TWO KEYS SIMULTANEOUSLY
WOULD EXIT YOU FROM THE LESSON.  BUT THE QUESTION ASKED  "WHICH KEY
IN ADDITION TO THE CTRL KEY"--THE CORRECT ANSWER WAS JUST THE C KEY.
%S YOU SHOULD HAVE ANSWERED BY TYPING THE C KEY.  REMEMBER THAT TO
ACTUALLY TYPE CTRL C YOU HOLD THE CTRL KEY DOWN, WHILE YOU PRESS THE
C KEY.  BUT YOU SHOULD NEVER USE THE CTRL C UNLESS YOU WANT TO
EXIT FROM THE LESSON.
WHAT WILL HAPPEN IF YOU ACTUALLY HOLD THE CTRL KEY DOWN AND
ALSO TYPE THE C KEY?
%R311 EXIT FROM LESSON
%R312 YOU WILL EXIT FROM THE LESSON
%R313 LESSON WILL END
%R314 THE LESSON WILL END
%W311 NOTHING
%& IF YOU TYPE THE WORD CTRL C NOTHING WILL HAPPEN TO THE
PROGRAM.  BUT IF YOU ACTULLY HIT THE TWO KEYS SIMULTANEOUSLY, YOU
WILL EXIT FROM THE LESSON.
%T04 REMEMBER THAT WHEN ANSWERING THE COMPUTER YOU SHOULD RESPOND
IN SHORT CONCISE ANSWERS.  DO NOT ADD EXTRA WORDS TO YOUR ANSWERS
OR THEY WILL BE COUNTED WRONG.  ANSWERS MUST MATCH EXACTLY.
IF YOU ADD AN EXTRA SPACE OR AN EXTRA WORD TO A CORRECT
ANSWER, IT WILL BE COUNTED WRONG.
Q4- WHEN RESPONDING TO A QUESTION, IF YOU MAKE ONE
SPELLING ERROR THAT YOU CAN NOT CORRECT, WILL YOUR
ANSWER STILL BE COUNTED CORRECT?
%R41 NO
%W41 YES
%& NO, IT WOULD BE COUNTED WRONG!!!  ALL ANSWERS MUST MATCH
EXACTLY.  IF YOU TYPE A ANSWER THAT IS COUNTED WRONG, AND YOU THOUGHT
IT WAS RIGHT---CHECK FOR SPELLING ERRORS AND OR EXTRA WORDS.
%T05 IF YOU MAKE THREE ERRORS IN A ROW, THE PROGRAM WILL
EXIT YOU FROM THE LESSON.  YOU WILL HAVE TO RESTART THE
LESSON.  SO IT IS VERY IMPORTANT THAT
YOU CAREFULLY READ EACH FRAME.
Q5- HOW MANY ERRORS IN A ROW ARE NECESSARY FOR THE
```

```
PROGRAM TO EXIT YOU FROM A LESSON?
%R51 3
%R52 THREE
%R53 THREE ERRORS
%R54 3 ERRORS
%R55 3 OR MORE
%R56 THREE OR MORE
%R57 OVER THREE
%R58 OVER 3
%W51 2
%& NO, IF YOU MAKE THREE ERRORS IN A ROW, YOU WILL
HAVE TO START THE LESSON OVER.
%T06 AFTER THE LAST FRAME IN A LESSON, YOUR SCORE WILL BE
PRINTED OUT ON THE TERMINAL.  YOUR SCORE IS ALSO
OUTPUT TO A FILE ON THE COMPUTER FOR THE CREATOR OF THE
LESSON TO USE IN REVISING AND EDITING THE LESSON.  AS SOON
AS YOUR SCORE IS DONE PRINTING OUT, THE LESSON IS OVER.
Q6- WHAT WILL BE PRINTED OUT FOLLOWING THE FINAL FRAME?
%R61 THE SCORE
%R62 SCORE
%R63 YOUR SCORE
%R64 MY SCORE
%W61 THE LESSON IS OVER
%& NO, AFTER THE LAST FRAME, YOUR SCORE WILL BE
PRINTED OUT FOR YOU.  AFTER THAT THE LESSON IS OVER.
%J
```

```
%T01 WELCOME TO LESSON TWO.  THIS LESSON WILL TEACH YOU
HOW TO WRITE YOUR OWN C.A.I. MODULES.  YOU NEED TO BE
FAMILIAR WITH A TEXT EDITOR SUCH AS DECS SYMBOLIC EDITOR
OR TECO.  YOU WILL NEED TO BE ABLE TO CREATE LESSON
FILES USING ONE OF THE TEXT EDITORS.
Q1- DO YOU KNOW A TEXT EDITOR WELL ENOUGH TO USE IT TO
CREATE A LESSON FILE?
%R11 YES
%W11 NO
%& DONT FEEL BAD, THE EDITOR IS NOT TOO HARD TO LEARN.
YOU SHOULD FIRST READ PAGES 5-11 TO 5-30 IN DIGITAL EQUIPMENTS
INTRODUCTION TO PROGRAMMING.  IT IS A NICE SUMMARY OF THE
SYMBOLIC EDITOR.  AFTER YOU HAVE READ IT, YOU SHOULD RUN
LESSON 3 WHICH IS A SUMMARY OF THE MOST IMPORTANT
COMMANDS THAT YOU WILL NEED TO CREATE LESSONS FOR CAI.
NOW YOU SHOULD HOLD THE CTRL KEY DOWN AND TYPE C.  THIS WILL EXIT
YOU FROM THIS LESSON.
%T02 TO CREATE A LESSON FOR CAI REQUIRES THAT YOU CREATE TWO
FILES.  ONE IS THE LESSON ITSELF AND THE OTHER IS A DIRECTORY
OF ALL LESSONS.  THE DIRECTORY IS CALLED CAILIB.FD.
IN CAILIB.FD YOU TYPE ALL OF THE LESSONS THAT ARE AVAILABLE
TO BE RUN.  THE LESSONS SHOULD BE LISTED MUCH LIKE THE
LESSON OPTIONS AT THE START OF THIS LESSON.  ALL YOU HAVE
TO TYPE IS THE LESSON NUMBERS AND TITLES.  THE PROGRAM WILL
DO THE REST!!!
Q2- WHAT IS THE NAME OF THE DIRECTORY FILE?
%R21 CAILIB.FD
%W21 CAILIB
%& CLOSE, BUT YOU FORGOT THE .FD EXTENSION AND
WITHOUT IT, THE COMPUTER WOULD NOT BE ABLE TO
FIND YOUR DIRECTORY.  THE CORRECT ANSWER WAS
CAILIB.FD
%W22 CAILIBFD
%& CLOSE, HOWEVER YOU LEFT OUT THE PERIOD BETWEEN THE
B AND THE F.  THE NAME OF THE DIRECTORY MUST BE EXACTLY
CAILIB.FD
%T03 THE NEXT FILE YOU CREATE WILL BE THE LESSON.  LESSONS ARE
NUMBERED SO THAT THE COMPUTER CAN DISTINGUISH THEM FROM ONE ANOTHER.
THEY ARE NAMED LESS(M).FD.  THE (M) IS A DIGIT. THE NUMBER
CORRESPONDS TO WHAT NUMBER THE LESSON IS.  FOR EXAMPLE,
IF I WERE CREATING MY FIRST LESSON, I WOULD NAME IT LESS1.FD
Q3- WHAT WOULD YOU NAME A LESSON THAT WAS TO BE THE FOURTH
LESSON THAT YOU CREATED?
%R31 LESS4.FD
%W31 LESSON 4.FD
%& NO, YOU MUST ABBREVIATE THE WORD LESSON.  IT MUST
BE EXACTLY IN THE FORM  LESS4.FD
%W32 LESSON4.FD
%& NO, YOU MUST SHORTEN LESSON TO LESS.  THE CORRECT
ANSWER WAS LESS4.FD
%W33 LESS4FD
%& YOU WERE CLOSE, BUT YOU LEFT OUT THE PERIOD
BETWEEN LESS4 AND THE FD EXTENSION.  THIS WOULD HAVE
CAUSED A PROBLEM.  THE CORRECT ANSWER WAS LESS4.FD
%W34 LESS4
%& NOT QUITE, YOU FORGOT THE EXTENSION.  THE
CORRECT ANSWER WAS LESS4.FD
%S WHEN CREATING FILES, IT IS VERY IMPORTANT THAT
YOU NAME YOUR FILES EXACTLY LIKE I SPECIFIED.  ALL
LESSONS MUST BE NAMED AS FOLLOWS.  LESS(M).FD.
WHERE THE (M) IS A NUMBER(DIGIT).  THEREFORE:
```

```
LESSON 1 WOULD BE LESS1.FD
LESSON 2 WOULD BE LESS2.FD
WHAT WOULD YOU NAME THE SIXTH LESSON?
%R311 LESS6.FD
%W311 LESSON6.FD
%& NO!!THE COMPUTER WOULD NOT ACCEPT THAT.
YOU SHOULD HAVE NAMED IT LESS6.FD
%T04 EACH LESSON CONSISTS OF A SERIES OF FRAMES.  EACH FRAME
CONSISTS OF THE FOLLOWING:
        1) TEXT AND A QUESTION
        2) POSSIBLE RIGHT ANSWERS
        3) POSSIBLE WRONG ANSWERS
        4) REMEDIATION OF MATCHED WRONG ANSWERS
        5) BRANCHING IF NO MATCHED ANSWER
FRAMES IN A LESSON ARE NUMBERED CONSECUTIVELY.
Q4- WHAT DOES THE FIRST PART OF A FRAME IN A LESSON CONSIST OF?
%R41 TEXT
%R42 TEXT AND A QUESTION
%R43 TEXT AND QUESTION
%R44 TEXT QUESTION
%W41 NUMBERED FRAMES
%& NO, THE QUESTION WANTED TO KNOW WHAT THE FIRST
PART OF A FRAME CONSISTED OF.  THE ANSWER WAS TEXT.
%S DONT BE CONCERNED IF YOU MISSED THAT ONE.  ALL I WAS
DOING WAS EXPLAINING THE GENERAL FLOW OF A CAI LESSON.
WITHIN EACH FRAME THERE IS TEXT, A QUESTION, POSSIBLE
RIGHT ANSWERS, POSSIBLE WRONG ANSWERS, REMEDIATION AND BRANCHING.
WHAT IS THE FIRST PART OF A FRAME IN A CAI LESSON?
%R411 TEXT
%R412 TEXT AND A QUESTION
%R413 TEXT, A QUESTION
%R414 TEXT AND QUESTION
%W411 A QUESTION
%& NO.  TEXT IS FIRST, THEN A QUESTION
%T05 TEXT CAN BE AS LONG AS NECESSARY.  THAT IS, IT CAN BE
MORE THAN ONE LINE.
THE ACTUAL TEXT MUST BE PRECEEDED BY THE FOLLOWING:
%T(#)SPACE
FOR EXAMPLE, IF I WERE STARTING THE FIRST FRAME
OF TEXT IN A LESSON, I WOULD TYPE:
%T01 WELCOME TO .....
THE PERCENT MUST BE FIRST, FOLLOWED BY THE LETTER T, THEN FOLLOWED
BY THE NUMBER OF THE FRAME, AND THEN BY A SPACE.  AFTER THE SPACE, YOU
MAY TYPE AS MUCH TEXT AS YOU HAVE FOR THAT FRAME.
Q5- WHEN BEGINNING EACH TEXT FRAME, YOU MUST FIRST TYPE A
PERCENT SIGN, FOLLOWED BY A NUMBER.  THEN BEFORE YOU TYPE IN TEXT
YOU MUST TYPE WHAT?
%R51 SPACE
%R52
%R53 A SPACE
%R54 A
%W51 NUMBER
%& NO, FIRST YOU TYPE  A PERCENT SIGN AND THEN THE FRAME
NUMBER AND THEN A SPACE.
%T06 TEXT IS TERMINATED BY A QUESTION AND A QUESTION MARK.
Q6- WHAT TERMINATES TEXT?
%R61 A QUESTION AND A QUESTION MARK
%R62 QUESTION AND QUESTION MARK
%R63 A QUESTION AND QUESTION MARK
%R64 QUESTION QUESTION MARK
%R65 A QUESTION AND ?
%R66 QUESTION AND ?
%R67 ?
%R68 A ?
%R69 QUESTION AND A ?
%W61 A QUESTION
```

```
%& THATS RIGHT, YOU MUST HAVE A QUESTION AT THE END OF TEXT, BUT
YOU MUST ALSO HAVE A QUESTION MARK.
%S ALL TEXT MUST BE FOLLOWED BY A QUESTION AND A QUESTION MARK.
THE QUESTION COVERS THE MATERIAL PRESENTED IN THE TEXT.  A QUESTION
MARK MUST FOLLOW THE QUESTION.
WHAT FOLLOWS THE QUESTION?
%R611 ?
%R612 QUESTION MARK
%R613 A QUESTION MARK
%W611 POSSIBLE RIGHT ANSWERS
%& NO, THE QUESTION MARK MUST BE BEFORE THE RIGHT ANSWERS.
%T07 CAUTION!!!  SINCE TEXT IS TERMINATED BY A QUESTION MARK, YOU
CAN NOT HAVE A QUESTION MARK IN TEXT.  YOU WILL HAVE TO REFER TO
IT AS I HAVE IN THIS FRAME, BY NAME.  THAT MEANS THAT YOU
WILL HAVE TO TYPE OUT THE WORDS QUESTION MARK.
Q7- CAN YOU HAVE A QUESTION MARK ITSELF IN THE TEXT PORTION
OF A FRAME?
%R71 NO
%W71 YES
%& NO, YOU MUST SPELL THE WORDS OUT IF YOU WISH TO
REFER TO A QUESTION MARK IN THE TEXT PORTION
OF A FRAME, ONLY.  THIS IS BECAUSE THE QUESTION MARK IS A
TERMINATOR FOR TEXT INPUT.
%T08 AS YOU LEARNED EARLIER, TEXT CAN BE AS MANY LINES AS NEEDED.
THE QUESTION CAN BE MORE THAN ONE LINE ALSO, AS LONG AS
IT IS FOLLOWED BY A QUESTION MARK.
Q8- WOULD A QUESTION THAT IS THREE LINES LONG BE
ACCEPTABLE AS LONG AS IT WAS FOLLOWED BY A QUESTION MARK?
%R81 YES
%W81 NO
%& OF COURSE IT WOULD BE ACCEPTABLE.  QUESTIONS CAN BE
ANY LENGTH.
%T09 SO FAR YOU HAVE WRITTEN SOME TEXT AND ASKED A QUESTION
ABOUT THE TEXT.  NEXT YOU WILL LIST ALL POSSIBLE RIGHT
ANSWERS.  MOST QUESTIONS WILL HAVE MORE THAN ONE RIGHT
POSSIBLE ANSWER.
Q9- WHAT FOLLOWS THE QUESTION AND QUESTION MARK?
%R91 RIGHT ANSWERS
%R92 ALL POSSIBLE RIGHT ANSWERS
%R93 POSSIBLE RIGHT ANSWERS
%R94 YOU LIST ALL POSSIBLE RIGHT ANSWERS
%R96 YOU WILL LIST ALL POSSIBLE RIGHT ANSWERS
%W91 ANSWERS
%& NOT QUITE, FIRST YOU LIST ALL POSSIBLE RIGHT
ANSWERS.  IT IS VERY IMPORTANT THAT YOU LIST ALL POSSIBLE RIGHT
ANSWERS IMMEDIATELY FOLLOWING THE QUESTION AND QUESTION MARK.
%W92 ALL POSSIBLE ANSWERS
%& FIRST ALL POSSIBLE RIGHT ANSWERS.
%T10 POSSIBLE RIGHT ANSWERS ARE NUMBERED SIMILAR TO THE
WAY TEXT WAS NUMBERED.  RIGHT ANSWERS ARE PRECEEDED
BY THE FOLLOWING
%R(##)SPACE
FOR EXAMPLE IF I WERE GOING TO WRITE THREE POSSIBLE
RIGHT ANSWERS FOR A QUESTION ON THE THIRD FRAME,
I WOULD DO IT AS FOLLOWS:
%R31 ANSWER 1
%R32 ANSWER 2
%R33 ANSWER 3
THE FIRST NUMBER AFTER THE %R REFERS TO THE FRAME NUMBER.  THE SECOND
NUMBER REFERS TO THE RIGHT ANSWER POSSIBILITY.
Q10-IF A POSSIBLE RIGHT ANSWER WAS LABELED  %R34  WHAT FRAME
WOULD IT BE?
%R101 THREE
%R102 3
%R103 FRAME THREE
%R104 FRAME 3
```

```
%R105 THIRD
%R106 THE THIRD FRAME
%R107 THIRD FRAME
%R108 NUMBER THREE
%R109 NUMBER 3
%W101 FOUR
%& NO, ITS LESSON THREE AND THE FOURTH POSSIBLE RIGHT ANSWER
%W102 4
%& NO, ITS LESSON THREE AND THE FOURTH POSSIBLE RIGHT ANSWER
%T11 A POSSIBLE RIGHT ANSWER CAN BE ONLY ONE LINE LONG.  YOU
MAY HAVE NINE POSSIBLE RIGHT ANSWERS.  RIGHT ANSWERS ARE LISTED
ON SEPERATE LINES, ONE AFTER ANOTHER.
Q11-HOW MANY POSSIBLE ONE LINE RIGHT ANSWERS CAN YOU HAVE?
%R111 9
%R112 NINE
%W111 SIXTY
%& NO, YOU MAY HAVE SIXTY CHARACTERS IN A SINGLE ANSWER.
BUT YOU CAN HAVE 9 POSSIBLE SIXTY CHARACTER RIGHT ANSWERS.
%T12 THE BASIC LOGIC THUS FAR, HAS BEEN TO WRITE A CONCISE
FRAME OF TEXT PRECEEDED BY %T#(SPACE) AND TERMINATED BY A
QUESTION COVERING THE TEXT AND A 'QUESTION.MARK'.  THEN ALL POSSIBLE
RIGHT ANSWERS TO THE QUESTION ARE LISTED, EACH PRECEEDED ON THE SAME
LINE BY  %R##(SPACE).  THEN ALL WRONG POSSIBLE ANSWERS ARE LISTED, EACH
ON A LINE, PRECEEDED ON THE SAME LINE BY  %W##(SPACE)
EACH WRONG POSSIBLE ANSWER IS FOLLOWED, ON THE NEXT LINE, BY
A LINE BEGINNING WITH %& FOLLOWED BY REMEDIAL TEXT THAT IS GEARED
TOWARD THE MISCONCEPTION CONVEYED BY THE CORRESPONDING WRONG ANSWER.
YOU MUST LIST AT LEAST ONE RIGHT ANSWER AND ONE WRONG ANSWER FOR
EACH FRAME.
Q12- IF YOU MUST HAVE AT LEAST ONE POSSIBLE WRONG ANSWER FOR EACH
FRAME, THEN WHAT WOULD THE MINIMUM NUMBER OF %& REMEDIAL MESSAGES
THAT YOU COULD HAVE PER FRAME?
%R121 1
%R122 ONE
%W121 NONE
%& NO THERE MUST BE ONE FOR EVERY POSSIBLE WRONG ANSWER
%W122 0
%& NO, THERE MUST BE ONE FOR EVERY POSSIBLE WRONG ANSWER.
%T13 FOR ALL QUESTIONS THERE WILL ALWAYS BE CERTAIN WRONG ANSWERS
THAT STUDENTS CONSISTENTLY GIVE.  YOU CAN LIST UP TO 9
OF THESE IMMEDIATELY FOLLOWING THE RIGHT POSSIBLE ANSWERS.
Q13- HOW MANY POSSIBLE WRONG ANSWERS CAN YOU HAVE PER FRAME?
%R131 9
%R132 NINE
%R133 UP TO 9
%R134 NINE OR LESS
%W131 MORE THAN NINE
%& NO, NINE IS THE MAXIMUM WRONG POSSIBLE ANSWERS PER FRAME.
%T14 POSSIBLE WRONG ANSWERS MUST BE PRECEEDED BY THE FOLLOWING:
%W(#)SPACE
THIS IS JUST LIKE THE FORMAT FOR THE POSSIBLE RIGHT ANSWERS.
IF I WERE LISTING THE FIRST POSSIBLE WRONG ANSWER
IN THE THIRD FRAME I WOULD DO IT AS FOLLOWS:
%W31 ANSWER
THE 3 REFERS TO THE THIRD FRAME AND THE 1 SIGNIFIES THE
FIRST POSSIBLE WRONG ANSWER TO BE LISTED FOR THE THIRD FRAME.
Q14- IF AN ANSWER WERE CODED  %W13  WHICH FRAME WOULD
YOU BE ON?
%R141 ONE
%R142 1
%R143 THE FIRST
%R144 FRAME ONE
%R145 FRAME 1
%R146 THE FIRST FRAME
%R147 FIRST
%W141 THREE
```

```
%& NO, YOU WOULD BE ON THE FIRST FRAME.  THE THREE SIGNIFIES
THAT IT IS THE THIRD WRONG POSSIBLE ANSWER FOR THE FIRST FRAME.
%W142 3
%& NO, IT WOULD BE THE THIRD POSSIBLE WRONG ANSWER
ON THE FIRST FRAME.
%T15 AFTER EACH WRONG POSSIBILITY, YOU INSERT A REMEDIAL
MESSAGE THAT CORRESPONDS TO THE ANTICIPATED WRONG ANSWER THAT IS
LISTED BEFORE IT.  SO WHEN TYPING A LESSON YOU WOULD ARRANGE
THE WRONG ANSWERS AS FOLLOWS:
%W11 RELY
%& NO, NOT QUITE, IT IS SPELLED RELAY.
%W12 .......
%& .....
ON THE LINE AFTER EACH WRONG POSSIBILITY, YOU TYPE  %& FOLLOWED BY
AS MUCH REMEDIATION AS NECESSARY.
EVERY %W POSSIBILITY MUST HAVE A %& FOLLOWING IT.
THE MESSAGE THAT YOU TYPE FOLLOWING THE %& CAN BE MORE THAN ONE
LINE LONG
Q15- WHAT MUST FOLLOW EVERY %W?
%R151 %&
%R152 A %&
%R153 A %& OPTION
%R154 %& FOLLOWED BY A REMEDIAL MESSAGE
%W151 REMEDIAL MESSGAE
%& THATS RIGHT, BUT FIRST THERE MUST BE A PERCENT
SIGN FOLLOWED BY A &.
%T16 THE & KEY IS TYPED BY PRESSING THE SHIFT KEY AND THE
6 KEY, AND IT IS THE CHARACTER THAT ALLOWS REMEDIAL
MESSAGES FOLLOWING EACH WRONG POSSIBILITY.  IT MUST BE
TYPED AT THE START OF THE LINE IMMEDIATELY FOLLOWING
EACH %W POSSIBILITY.
Q16- WHAT KEY ALLOWS REMEDIAL MESSAGES FOLLOWING EACH
POSSIBLE WRONG ANSWER?
%R161 &
%R162 SHIFT 6
%R163 %&
%W161 6
%& NO, ITS SHIFT 6.  YOU TYPED THE 6 BUT FORGOT
TO HOLD THE SHIFT KEY DOWN WHILE YOU WERE PRESSING IT.
%T17 IF A STUDENT GIVES AN ANSWER THAT IS CORRECT, THE PROGRAM
GOES TO THE NEXT FRAME.  IF THE STUDENT GIVES AN ANSWER THAT
IS WRONG, BUT MATCHES ONE OF THE POSSIBLE WRONG ANSWERS THAT YOU
HAVE LISTED, THEN THE %& PRINTS OUT THE CORRESPONDING
REMEDIAL MESSAGE AND THEN THE PROGRAM GOES TO THE NEXT FRAME.
HOWEVER, IF A STUDENT GIVES AN ANSWER THAT DOES NOT MATCH ONE
OF THE POSSIBLE ANSWERS THAT YOU LISTED, THE PROGRAM WILL
JUST PRINT OUT WRONG!!.  THERE IS AN OPTION AVAILABLE TO HANDLE
THIS SITUATION.  IT IS THE %S OPTION.  IT ALLOWS YOU TO WRITE ADDITIONA

TEXT AND QUESTIONS WHICH WILL ONLY BE PRINTED WHEN THE
STUDENTS ANSWER DOES NOT MATCH ANY OF THE ANSWERS LISTED.
THE %S IS OPTIONAL, UNLIKE THE %&.  YOU MAY USE %S
ON EVERY FRAME, ON DIFFICULT FRAMES OR NOT AT ALL.
Q17- IF A STUDENTS ANSWER MATCHES ONE OF THE WRONG
ANSWERS THAT YOU LISTED, WILL THE %S OPTION BE PRINTED?
%R171 NO
%W171 YES
%& NO, PERCENT S IS ACTIVATED ONLY BY AN ANSWER FROM THE
STUDENT THAT DOES NOT MATCH ANY OF THE ONES THAT YOU LISTED.
%T18 AFTER YOU INSERT A %S, YOU HAVE BEGUN WHAT IS KNOWN AS
BRANCHING.  BRANCHING MEANS WRITING TEXT THAT WILL BE PRESENTED IN
ONE CASE AND SKIPPED IN ANOTHER.  IF A STUDENT GETS A QUESTION
RIGHT OR IF HE MATCHES ONE OF THE POSSIBLE WRONG ANSWERS THAT YOU
LISTED, THE TEXT FOLLOWING %S WILL NOT BE PRESENTED.  BUT IF A
STUDENT GIVES AN ANSWER THAT WAS NOT ONE OF THE ONES THAT YOU LISTED
THEN THE TEXT FOLLOWING THE %S WILL BE PRESNTED.
```

Q18- IF YOU USE THE %S OPTION, AND A STUDENT GIVES
THE CORRECT ANSWER TO THE QUESTION, WILL THE %S TEXT BE PRESENTED?
%R181 NO
%W181 YES
%& NO, THE %S OPTION TEXT IS ONLY PRESENTED IF THE STUDENTS ANSWER
DOES NOT MATCH ANY OF THE RIGHT OR WRONG POSSIBLE ANSWERS
THAT YOU LISTED.
%T19 THE FORMAT OF THE S OPTION IS AS FOLLOWS. THE FIRST THING
THAT YOU TYPE IS A %S. YOU FOLLOW THE %S WITH REMEDIAL TEXT
COVERING THE MATERIAL PRESENTED IN THE LAST FRAME. THE PURPOSE OF
THE S OPTION IS TO CLARIFY ANY FRAME THAT COULD POSSIBLY BE SO
CONFUSING THAT A STUDENT MIGHT GIVE AN UNEXPECTED ANSWER.
YOU MUST END THE REMEDIAL TEXT WITH A QUESTION AND A QUESTION
MARK. AFTER THAT YOU LIST POSSIBLE RIGHT AND WRONG ANSWERS IN THE
SAME MANNER AS YOU WOULD IN A REGULAR FRAME. IF A STUDENTS ANSWER
DOES NOT MATCH ANY OF THE RIGHT AND WRONG ANSWERS THEN THE PROGRAM
CHECKS FOR ANOTHER %S. IF THERE IS ANOTHER %S THEN THE
PROGRAM INPUTS TEXT AND THE ENTIRE %S SEQUENCE IS REPEATED.
IF THERE IS NOT ANOTHER %S THEN THE PROGRAM GOES TO THE NEXT FRAME.
Q19- IF A STUDENT GIVES AN UNMATCHED WRONG ANSWER TO A QUESTION,
AND THERE IS NOT ANOTHER %S OPTION, WHERE DOES THE PROGRAM GO?
%R191 NEXT FRAME
%R192 NEXT REGULAR FRAME
%R193 THE NEXT FRAME
%R194 NEXT REGULAR FRAME
%R195 THE NEXT REGULAR FRAME
%W196 NEXT PERCENT S OPTION
%& THE PROGRAM WOULD FIRST CHECK FOR THE NEXT PERCENT S OPTION
BUT IF THERE WAS NOT ANOTHER ONE, THEN THE PROGRAM WOULD GO ON TO
THE NEXT REGULAR FRAME.
%T20 HERE IS AN EXAMPLE OF WHAT THE %S OPTION WOULD LOOK
LIKE AFTER YOU WROTE A FRAME USING IT.

%R35 RELAY
%W31 RELY
%$ YOU MISSPELLED THE WORD, IT IS RELAY
%S THIS WOULD BE TEXT THAT YOU WOULD WRITE TO HELP CLARIFY
THE PRECEEDING FRAME. TEXT IS HALTED BY A QUESTION AND A
QUESTION MARK.
%R311 RIGHT ANSWERS LISTED
%W311 WRONGS LISTED
%& EACH FOLLOWED BY THIS

AS THE EXAMPLE ILLUSTRATED, A %S "FRAME" LOOKS AND IS
WRITTEN JUST LIKE A REGULAR FRAME.
IF THE STUDENTS ANSWER MATCHS ONE OF THE LISTED ANSWERS IN THE
S OPTION THEN THE PROGRAM RETURNS TO THE NEXT REGULAR FRAME
Q20- WHEN WRITING A %S OPTION, ARE THE POSSIBLE RIGHT ANSWERS OR
THE POSSIBLE WRONG ANSWERS LISTED FIRST?
%R201 RIGHT
%R202 RIGHT ANSWERS
%R203 THE RIGHT ANSWERS
%R204 THE POSSIBLE RIGHT ANSWERS
%R205 THE POSSIBLE RIGHT ANSWERS ARE LISTED FIRST
%R206 POSSIBLE RIGHT ANSWERS
%W201 WRONG
%& NO, JUST AS IN REGULAR FRAMES ALL RIGHT POSSIBLE
ANSWERS ARE LISTED FIRST.
%W202 WRONG ANSWERS
%& NO, JUST AS IN A REGULAR FRAME, ALL RIGHT POSSIBLE
ANSWERS ARE LISTED FIRST
%T21 IF THE STUDENTS ANSWER TO THE QUESTION IN THE %S OPTION
DOES NOT MATCH ANY OF THE ANSWERS LISTED, THEN ONE OF TWO
THINGS CAN HAPPEN. IF THERE IS ANOTHER %S OPTION, YOU CAN USE AS
MANY IN A ROW AS YOU NEED, THEN THE TEXT ETC. FOLLOWING IT WILL BE
PRINTED. IF THERE IS NOT ANOTHER %S OPTION THEN THE PROGRAM RETURNS

```
TO THE NEXT REGULAR SCHEDULED FRAME.
Q21- CAN YOU HAVE MORE THAN ONE %S OPTION PER FRAME, IE-IN
A ROW?
%R211 YES
%W211 NO
%& THE ANSWER WAS YES.  IF AFTER BRANCHING TO A PERCENT
S OPTION, A CORRECT ANSWER BY THE STUDENT AUTOMATICALLY
RETURNS THE STUDENT TO THE NEXT FRAME.  BUT IF NO MATCH
OCCURS THEN IF YOU HAD USED ANOTHER PERCENT OPTION IT WOULD HAVE
BEEN PRINTED.
%T22 YOU CAN HAVE AS MANY FRAMES IN A LESSON AS YOU NEED.
BUT THE COMPUTER WILL NEED TO KNOW WHICH FRAME YOU INTEND
TO BE THE LAST ONE.  TO SIGNIFY THE LAST FRAME, YOU
TYPE A %J ON THE LINE FOLLOWING THE LAST
LINE IN YOUR LESSON.
Q22- ON WHICH LINE IS THE %J TYPED?
%R221 THE LAST
%R222 THE LAST LINE
%R223 ON THE LAST LINE
%R224 ON THE LINE FOLLOWING THE LAST LINE IN YOUR LESSON
%R225 LAST
%W221 THE LAST LINE OF THE LESSON
%&NO, IT GOES AFTER THE LAST LINE IN THE LESSON
%W222 THE LAST LINE IN YOUR LESSON
%& NO, IT GOES AFTER THE LAST LINE IN THE LESSON
%T23 NOW YOU SHOULD BE READY TO WRITE YOUR OWN CAI LESSONS.
HERE ARE TWO THINGS THAT WILL HELP YOU.
        1) READ THE CAI MANUAL AVAILABLE FROM TOM BURNE.
        2) AFTER YOU EXIT FROM THIS LESSON, TYPE THE FOLLOWING:
           TYPE LESS4.FD
           AND THEN PRESS CARRIAGE RETURN.
LAST QUESTION!!!- DID YOU FIND THIS LESSON:
CONFUSING,AWKWARD,BORING,TOO EASY,TOO HARD,INSTRUCTIONAL
OR NONE OF THE ABOVE?
%R231 CONFUSING
%R232 BORING
%R233 AWKWARD
%R234 TOO EASY
%R235 TOO HARD
%R236 INSTRUCTIONAL
%W231 NONE OF THE ABOVE
%& I MIGHT AGREE WITH YOU!!!
%J
```

```
%T01 TO WRITE A CAI LESSON YOU MUST KNOW HOW TO USE A TEXT
EDITOR PROGRAM SUCH AS THE SYMBOLIC EDITOR.  THE SYMBOLIC EDITOR
IS ONE OF THE SIMPLEST TEXT EDITORS TO LEARN AND USE.  A TEXT
EDITOR ALLOWS YOU TO INPUT TEXT INTO THE COMPUTER VIA THE
KEYBOARD(DEC WRITER).
Q1- WHAT DOES THE SYMBOLIC EDITOR ALLOW YOU TO DO?
%R11 INPUT TEXT
%R12 INPUT TEXT INTO THE COMPUTER
%R13 TYPE IN TEXT
%R14 TYPE TEXT
%R15 TYPE CAI LESSONS
%R16 TYPE LESSONS
%R17 INPUT LESSONS
%R18 WRITE CAI LESSONS
%R19 WRITE A CAI LESSON
%W11 LEARN AND USE
%& YOU WILL LEARN AND USE THE SYMBOLIC EDITOR SILLY.  BUT
THE SYMBOLIC EDITOR WILL ENABLE YOU TO INPUT TEXT INTO THE COMPUTER
AND THUS WRITE CAI LESSONS.
%T02 TELETYPE INPUT IS USED TWO WAYS BY THE EDITOR.  COMMANDS TO
THE EDITOR MUST BE DIFFERENTIATED FROM TEXT.  FOR THIS REASON THERE
ARE TWO MODES OF USE:  COMMAND MODE AND TEXT MODE.  COMMAND
MODE ALLOWS YOU TO GIVE THE COMPUTER INSTRUCTIONS AND TEXT MODE
ACTUALLY ALLOWS YOU TO TYPE IN TEXT.
Q2- WHICH MODE WOULD YOU WANT TO BE IN TO TYPE IN THE TEXT
OF A LESSON?
%R21 TEXT MODE
%R22 TEXT
%W21 COMMAND
%& NO, YOU WOULD USE COMMAND MODE TO TAKE YOU TO TEXT MODE
WHICH IS WHERE YOU WOULD WANT TO BE TO TYPE IN TEXT.
%T03 WHEN YOU START THE EDITOR, YOU ARE INITIALLY IN COMMAND MODE.
ANY ONE LETTER IS INTERPRETED AS A COMMAND.  FOR INSTANCE THE
FOLLOWING COMMANDS WOULD LIST PORTIONS OF THE TEXT BUFFER.
        L  (RETURN)       LISTS ENTIRE BUFFER
        5L (RETURN)       LISTS ONLY LINE 5
        5,7L  (RETURN)    LISTS LINES 5,6 AND 7
Q3- WHAT LINES WOULD BE LISTED BY THE FOLLOWING COMMAND:
16,21L (RETURN)?
%R31 LINES 16 THRU 21
%R32 16,17,18,19,20, AND 21
%R33 16,17,18,19,20,21
%R34 LINES 16,17,18,19,20, AND 21
%R35 LINES 16 TO 21
%R36 16 TO 21
%R37 16 THRU 21
%W31 LINES 16 AND 21
%& NO, LINES 16,17,18,19,20 AND 21 WOULD HAVE BEEN
LISTED BY THE COMMAND  16,21L.
%W12 16,21
%& IT WOULD LIST LINES 16 THRU 21
%T04 CERTAIN COMMANDS WILL AUTOMATICALLY PUT YOU IN TEXT MODE.
THAT MEANS ANYTHING THAT YOU TYPE FOLLOWED BY A RETURN WILL BE
ENTERED INTO A TEXT BUFFER.  THESE COMMANDS ARE:
        A     APPEND TO THE TEXT BUFFER, AFTER THE LAST LINE OF TEXT
        I     INSERT INTO TEXT BUFFER BEFORE SPECIFIED LINE
              (E.G. 5I INSERTS TEXT BEFORE LINE 5)
              ('I' BY ITSELF, INSERTS TEXT BEFORE LINE 1)
        C     CHANGE A CERTAIN LINE
              ('4C' DELETES LINE 4 AND ALLOWS YOU TO RETYPE IT)
```

```
TO TERMINATE TEXT MODE AND RETURN TO COMMAND MODE, YOU TYPE
        CTRL L (CTRL FORM)    OR    CTRL G (CTRL BELL)
Q4- TYPING AN "A" WILL PUT YOU IN WHAT MODE?
%R41 TEXT
%R42 TEXT MODE
%W41 COMMAND
%& NO, TYPING AN "A" WOULD PUT YOU IN TEXT MODE.
%W42 COMMAND MODE
%& NO, TYPING AN A WOULD HAVE PUT YOU IN TEXT MODE
%T05 THERE ARE TWO SPECIAL CHARACTERS THAT YOU WILL NEED TO KNOW
ABOUT.  ONE IS THE RETURN.  CARRIAGE RETURN IS THE TERMINATOR
FOR TEXT LINES AND FOR COMMANDS.  THIS MEANS THAT ONLY LINES
THAT ARE FOLLOWED BY RETURN ARE ENTERED INTO THE TEXT BUFFER.
LIKEWISE, COMMANDS ARE NOT EXECUTED UNTIL RETURN IS TYPED.
Q5- WHICH KEY IS THE TERMINATOR FOR COMMANDS?
%R51 RETURN
%R52 THE RETURN
%R53 THE CARRIAGE RETURN
%R55 CARRIAGE RETURN
%R56 CR
%R57 C R
%R58
%W51 CTRL L
%& NO, THE CARRIAGE RETURN IS THE TERMINATOR FOR COMMANDS.
%T06 THERE ARE TWO KEYS THAT YOU MAY USE TO CANCEL TEXT OR COMMANDS.
THE FIRST ONE IS THE CTRL/U KEY.  IT CAN BE USED BY HOLDING THE CTRL
KEY DOWN, WHILE PRESSING THE U KEY.  WHEN IN COMMAND MODE, TO CANCEL A
COMMAND WITH THE CTRL/U, ALL YOU HAVE TO DO IS TO PRESS THE RUBOUT KEY
ONCE.  THE COMPUTER WILL TYPE A QUESTION MARK AND WILL BE READY TO
ACCEPT A NEW COMMAND.  WHEN IN TEXT MODE, TO CANCEL THE ENTIRE CURRENT
LINE OF TEXT AND LET YOU CONTINUE AS IF THE LINE NEVER HAPPENED,
YOU TYPE CTRL/U.
Q6- WHAT MODE WOULD YOU HAVE TO BE IN TO USE CTRL/U TO CANCEL
A COMMAND?
%R61 COMMAND
%R62 COMMAND MODE
%W61 TEXT MODE
%& NO, TO BE ISSUING A COMMAND, YOU MUST BE IN COMMAND MODE.
%W62 TEXT
%& NO, TO BE ISSUING A COMMAND, YOU MUST BE IN COMMAND MODE.
%T07 THE OTHER KEY THAT CAN BE USED TO CANCEL COMMANDS OR TEXT
IS THE RUBOUT.  IN THE EDITOR, A RUBOUT WILL CANCEL A COMMAND IF
YOU TYPE IT BEFORE THE RETURN.  AND AS WITH THE CTRL/U IT LEAVES
YOU READY TO ISSUE ANOTHER COMMAND. IN TEXT MODE, A RUBOUT WILL
DELETE THE LAST CHARACTER THAT YOU HAVE TYPED.  IT ALLOWS YOU
TO DELETE CHARACTERS ALL THE WAY BACK TO THE FIRST OF THE LINE.
Q7- IF I WERE EDITING TEXT AND I PRESSED THE RUBOUT KEY THREE
TIMES IN A ROW ON THE SAME LINE, WHICH MODE WOULD I HAVE TO BE IN?
%R71 TEXT
%R72 TEXT MODE
%R73 THE TEXT MODE
%W71 COMMAND
%& NO, AS SOON AS YOU PRESS RUBOUT IN COMMAND MODE, THE COMMAND
THAT YOU WERE TYPING IS ERASED AND THE COMPUTER WILL TYPE A
QUESTION MARK AND RETURN AND BE READY FOR A NEW COMMAND.  BUT IN TEXT
MODE YOU MAY ERASE FOUR CHARACTERS OR UP TO A WHOLE LINE OF
TEXT WITH THE RUBOUT KEY.
%W72 COMMAND MODE
%& NO, IN COMMAND MODE AS SOON AS YOU PRESS RUBOUT THE COMPUTER
RETURNS AND PRINTS A QUESTION MARK.  IN TEXT MODE YOU MAY ERASE
FROM 1 CHARACTER UP THE FIRST ONE TYPED ON THE LINE THAT YOU ARE ON.
%T08 THE SYMBOLIC EDITOR IS AN EDITOR PROGRAM THAT CENTERS
AROUND NUMBERED LINES.  THE EDITOR ASSIGNS A LINE NUMBER TO
EVERY LINE(IMPLICITLY-IE. IT DOES IT AUTOMATICALLY).
IN COMMAND MODE, TO HAVE THE CURRENT LINE PRINTED, YOU TYPE
. (PERIOD) FOLLOWED BY A RETURN.  TO HAVE THE LAST LINE OF TEXT
```

PRINTED YOU TYPE / (SLASH) FOLLOWED BY A RETURN.
IT IS ALSO POSSIBLE TO BE ABLE TO PRINT OTHER LINES
BY USING THESE AS A REFERENCE FOR RELATIVE LINE SPECIFICATIONS
WITHOUT HAVING TO FIND OUT THEIR ABSOLUTE LINE NUMBERS.
AS AN EXAMPLE,
     /-5L   MEANS PRINT THE LINE THAT IS 5TH FROM THE BOTTOM LINE
Q8- TO PRINT A LINE THAT IS THE CURRENT LINE BUT NOT THE ACTUAL
LAST OR FINAL LINE OF TEXT, WHAT WOULD YOU TYPE(ASSUMING YOU ARE
IN COMMAND MODE)?
%R81 .L
%R82 .L FOLLOWED BY A RETURN
%R83 .L(RETURN)
%R84 .L RETURN
%R85 . L
%R86 . L(RETURN)
%W81 /L
%& /L WOULD HAVE LISTED THE LAST LINE IN THE TEXT BUFFER. IF YOU
WERE NOT ON THAT LINE, YOU WOULD THEN HAVE TO HAVE TYPED .L (RETURN).
%T09 THERE IS NO REASON NOT TO KNOW THE CURRENT LINE NUMBER.
THERE ARE TWO KEYS THAT ARE USED FOR THIS. THE = (EQUALS KEY)
AND THE : (COLON KEY). THEY WORK IN A SIMILAR MANNER AS THE FOLLOWING
EXAMPLES ILLUSTRATE,
     .=   PRINTS CURRENT LINE NUMBER
     .:   PRINTS CURRENT LINE NUMBER
     /:   PRINTS NUMBER OF THE LAST(FINAL) LINE
     /=   PRINTS NUMBER OF THE LAST(FINAL) LINE
ONE THING YOU WILL OFTEN WANT TO DO WITH THE EDITOR IS TO
EXAMINE THE PREVIOUS LINE OR THE FOLLOWING LINE. THIS MUST BE DONE
IN COMMAND MODE. TO DO THIS YOU WOULD TYPE
     .-1L   OR
     .+1L
Q9- TO PRINT THE PREVIOUS LINE WHAT WOULD YOU TYPE?
%R91 .-1L
%R92 <
%R93 SHIFT ,
%W91 .+1L
%& THAT WOULD LIST THE VERY NEXT LINE, TO PRINT THE PREVIOUS LINE YOU
WOULD TYPE .-1L.
%T10 TO ENTER TEXT INTO THE COMPUTER VIA THE TELETYPE, YOU
WOULD TYPE "A (RETURN)". THIS WOULD ADD TEXT ONTO THE END OF
AN EXISTING BUFFER OR AT THE START IF THE BUFFER IS EMPTY.
FOLLOWING TYPING "A (RETURN)" ANYTHING THAT YOU TYPE THAT TERMINATES
WITH A RETURN IS ENTERED INTO THE BUFFER, UNTIL YOU TYPE A CTRL/L
OR A CTRL/G TO RETURN TO COMMAND MODE.
Q10- WHEN IN COMMAND MODE, WHAT DO YOU TYPE TO ALLOW YOU TO
TO ENTER TEXT?
%R101 A (RETURN)
%R102 A FOLLOWED BY RETURN
%R103 A
%W101.CTRL/G
%& NO, THE CORRECT ANSWER WAS THE KEY A FOLLOWED BY RETURN
%T11 IF YOU WANT TO INSERT TEXT BEFORE A LINE, YOU USE THE I COMMAND.
FOR EXAMPLE THE COMMAND 26I (RETURN) WOULD ALLOW YOU TO INSERT TEXT
BEFORE THE 26TH LINE.THE I COMMAND WITH NO NUMBER PRECEEDING IT, WILL
INSERT TEXT BEFORE THE FIRST LINE OF THE CURRENT TEXT BUFFER.
Q11- TO INSERT TEXT BEFORE THE 3RD LINE OF THE CURRENT BUFFER, WHAT
WOULD YOU TYPE?
%R111 3I
%R112 3I (RETURN)
%R113 3I FOLLOWED BY RETURN
%W111 A
%& THAT WOULD ALLOW YOU TO ENTER TEXT ALRIGHT, EXCEPT THE TEXT
WOULD BE PLACED AT THE END OF THE CURRENT TEXT BUFFER.
%T12 SOMETIMES A LINE IS IN THE WRONG PLACE OR IS FULL OF MANY ERRORS
AND YOU WANT TO DELETE IT, OR PERHAPS A GROUP OF LINES. TO DO THIS
YOU NEED TO USE THE D COMMAND. THE FOLLOWING EXAMPLE SHOWS HOW TO

```
USE THE D COMMAND,
     6D(RETURN)      DELETES LINE 6
     4,7D(RETURN)      DELETES LINES 4,5,6 AND 7
IN THE FIRST EXAMPLE, THE OLD LINE 6 IS NOW GONE, AND ALL
SUBSEQUENT LINES HAVE NOW BEEN RENUMBERED.  IN THE SECOND
CASE, LINES 4,5,6 AND 7 HAVE BEEN DELETED.  AND THE
OLD LINE 8 WOULD NOW BE LINE 4, THRU IMPLICIT NUMBERING BY
THE COMPUTER.
Q12- GIVE A SINGLE COMMAND TO DELETE FROM LINE 19 THRU LINE 37
OF A PROGRAM?
%R121 19,37D
%R122 19,37D(RETURN)
%R123 19,37D FOLLOWED BY RETURN
%W121 19,37 D
%& YOU WERE CLOSE, BUT YOU SHOULD NOT HAVE ADDED THE SPACE
BETWEEN THE 7 AND THE D.
%T13 SOMETIMES THE SIMPLEST WAY OF CORRECTING A LINE THAT
HAS MISTAKES IN IT, IS TO DELETE THE LINE AND TYPE A REPLACEMENT
FOR IT.  THE "C" COMMAND IS ABLE TO ACCOMPLISH THESE TWO
OBJECTIVES.  YOU ARE IN TEXT MODE AFTER YOU TYPE A "C"
COMMAND UNTIL YOU TYPE A CTRL/G OR A CTRL/L.
Q13- TO USE THE "C" COMMAND DO YOU HAVE TO BE IN TEXT
OR COMMAND MODE?
%R131 COMMAND
%R132 COMMAND MODE
%W131 TEXT
%& NO, TO GIVE ANY COMMAND REQUIRES YOU TO BE IN COMMAND MODE.
%W132 TEXT MODE
%& NO, TO GIVE ANY COMMAND REQUIRES YOU TO BE IN COMMAND MODE.
%T14 THE S (SEARCH) COMMAND IS ONE OF THE MOST POWERFUL AND USEFUL
COMMANDS IN THE EDITOR FOR CORRECTING ERRORS WITHIN A LINE.
"S" IS FOLLOWED BY RETURN, AND THEN BY THE CHARACTER TO BE SEARCHED.
THEN THE TTY(DECWRITER) PRINTS THE PORTION OF THE LINE LEADING
UP TO AND INCLUDING THE SEARCH CHARACTER.  IF A SEARCH
CHARACTER IS NOT FOUND, THEN THE LINE IS NOT CHANGED.
SEVERAL THINGS MAY HAPPEN ONCE THE CHARACTER HAS BEEN FOUND IN THIS
WAY, DEPENDING ON THE KEY THAT THE USER TYPES NEXT:
     KEY                      WHAT HAPPENS
     ---                      ---- -------
     1.  CTRL/U            ENTIRE PRINTED PORTION OF LINE IS DELETED
     2.  RETURN            ENTIRE UNPRINTED PORTION OF LINE
                           DELETED (LINE NOW ENDS AT SEARCH CHAR.)
     3.  RUBOUT            DELETES SINGLE CHARACTER TO THE LEFT
                           OF THE SEARCH CHARACTER
     4.  ANY CHARACTER
         FROM KEYBOARD     INSERTS CHARACTERS IMMEDIATELY
                           AFTER THE SEARCH CHARACTER
     5.  LINE FEED         DIVIDES THE LINE INTO 2 LINES
                           AFTER THE SEARCH CHARACTER
     6.  CTRL/L            CONTINUE SEARCHING TO NEXT OCCURANCE
                           OF THE SEARCH CHARACTER
     7.  CTRL/G            CHANGE THE SEARCH CHARACTER TO THE
                      NEXT ONE TYPED, AND SEARCH FOR IT
THE ENTIRE BUFFER MAY BE SEARCHED BY TYPING
     S
     (SEARCH CHARACTER)
A SINGLE LINE MAY BE SEARCHED, AS IN
     6S
     (SEARCH CHARACTER)
A BLOCK OF LINES MAY BE SEARCHED, AS IN
     40,50S
     (SEARCH CHARACTER)
THE FOLLOWING QUESTIONS, QUESTIONS 14 AND 15, WILL REFER TO THE
NEXT LINE
        ABCDEFGHIJKLMN
YOU WILL BE ASKED TO TYPE THE WAY THE LINE WILL LOOK WHEN
```

```
YOU HAVE EDITED IT IN THE MANNER DESCRIBED IN THE QUESTIONS.
Q14- IF I DO THE FOLLOWING:
     .S(RETURN)
     G
     (RETURN)
WHAT IS THE LINE GOING TO LOOK LIKE?
%R141 ABCDEFG
%W141 ABBCDEF
%& NOT QUITE, THE CORRECT ANSWER WOULD HAVE BEEN
ABCDEFG
BECAUSE THE SEARCH WOULD HAVE PRINTED UP TO AND
INCLUDING THE G.
%T15 THIS FRAME ONLY CONSISTS OF A QUESTION COVERING THE LINE:
        ABCDEFGHIJKLMN
Q15- WHAT WILL THE LINE LOOK LIKE IF I SEARCH FOR Z?
%R151 ABCDEFGHIJKLMN
%R152 THE SAME
%R153 SAME
%R154 IDENTICAL
%R155 UNCHANGED
%W151 ABCDEFGHIJKLMNZ
%& NO, THE LINE DOES NOT CHANGE IF YOU SEARCH FOR
A CHARACTER THAT IS NOT CONTAINED IN THAT LINE.
%J
```