DOCUMENT RESUME

ED 067 870 EM 010 339

AUTHOR Friend, Jamesine E.; And Others
TITLE Student Performance in Computer-Assisted Instruction in Programming.
INSTITUTION Stanford Univ., Calif. Inst. for Mathematical Studies in Social Science.
SPONS AGENCY Office of Naval Research, Washington, D.C. Personnel and Training Research Programs Office.
PUB DATE 10 May 72
NOTE 96p.; Psychology and Education Series; Technical Report No. 184

EDRS PRICE MF-$0.65 HC-$3.29
DESCRIPTORS College Students; *Computer Assisted Instruction; *Computer Science Education; Program Evaluation; *Programing; Programing Languages
IDENTIFIERS AID; *Algebraic Interpretive Dialogue

ABSTRACT

A computer-assisted instructional system to teach college students the computer language, AID (Algebraic Interpretive Dialogue), two control programs, and data collected by the two control programs are described. It was found that although first response errors were often those of AID syntax, such errors were easily corrected. Secondly, while students easily learned the mechanics of the instructional system, they rarely used features that allowed student control. Thirdly, algebraic formulations of problems appeared to be more difficult than transforming algebraic expressions into AID commands. Finally, the students had the greatest difficulty understanding hierarchy of arithmetic operations, use of functions, and the execution sequence of AID commands. (Author/RH)

STUDENT PERFORMANCE IN COMPUTER-ASSISTED INSTRUCTION IN PROGRAMMING

BY

J. E. FRIEND, J. D. FLETCHER and R. C. ATKINSON

TECHNICAL REPORT NO. 184

MAY 10, 1972

PSYCHOLOGY & EDUCATION SERIES

This research was sponsored by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract No. N00014-67-A-0112-0054, Contract Authority Identification Number, NR No. 154-326.

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

TECHNICAL REPORTS

PSYCHOLOGY SERIES

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

(Place of publication shown in parentheses; if published title is different from title of Technical Report, this is also shown in parentheses.)

(For reports no. 1 - 44, see Technical Report no. 125.)

50 R. C. Atkinson and R. C. Calfee. Mathematical learning theory. January 2, 1963. (In B. B. Wolman (Ed.), Scientific Psychology. New York: Basic Books, Inc., 1965. Pp. 254-275)

51 P. Suppes, E. Crothers, and R. Weir. Application of mathematical learning theory and linguistic analysis to vowel phoneme matching in Russian words. December 28, 1962.

52 R. C. Atkinson, R. Calfee, G. Sommer, W. Jeffrey and R. Shoemaker. A test of three models for stimulus compounding with children. January 29, 1963. (J. exp. Psychol., 1964, 67, 52-58)

53 E. Crothers. General Markov models for learning with inter-trial forgetting. April 8, 1963.

54 J. L. Myers and R. C. Atkinson. Choice behavior and reward structure. May 24, 1963. (Journal math. Psychol., 1964, 1, 170-203)

55 R. E. Robinson. A set-theoretical approach to empirical meaningfulness of measurement statements. June 10, 1963.

56 E. Crothers, R. Weir and P. Palmer. The role of transcription in the learning of the orthographic representations of Russian sounds. June 17,. 1963.

57 P. Suppes. Problems of optimization in learning a list of simple items. July 22, 1963. (In Maynard W. Shelly, II and Glenn L. Bryan (Eds.), Human Judgments and Optimality. New York: Wiley. 1964. Pp. 116-126)

58 R. C. Atkinson and E. J. Crothers. Theoretical note: all-or-none learning and intertrial forgetting. July 24, 1963.

59 R. C. Calfee. Long-term behavior of rats under probabilistic reinforcement schedules. October 1, 1963.

60 R. C. Atkinson and E. J. Crothers. Tests of acquisition and retention, axioms for paired-associate learning. October 25, 1963. (A comparison of paired-associate learning models having different acquisition and retention axioms, J. math. Psychol., 1964, 1, 285-315)

61 W. J. McGill and J. Gibbon. The general-gamma distribution and reaction times. November 20, 1963. (J. math. Psychol., 1965, 2, 1-18)

62 M. F. Norman. Incremental learning on random trials. December 9, 1963. (J. math. Psychol., 1964, 1, 336-351)

63 P. Suppes. The development of mathematical concepts in children. February 25, 1964. (On the behavioral foundations of mathematical concepts. Monographs of the Society for Research in Child Development, 1965, 30, 60-96)

64 P. Suppes. Mathematical concept formation in children. April 10, 1964. (Amer. Psychologist, 1966, 21, 139-150)

65 R. C. Calfee, R. C. Atkinson, and T. Shelton, Jr. Mathematical models for verbal learning. August 21, 1964. (In N. Wiener and J. P. Schoda (Eds.), Cybernetics of the Nervous System: Progress in Brain Research. Amsterdam, The Netherlands: Elsevier Publishing Co., 1965. Pp. 333-349)

66 L. Keller, M. Cole, C. J. Burke, and W. K. Estes. Paired associate learning with differential rewards. August 20, 1964. (Reward and information values of trial outcomes in paired associate learning. (Psychol. Monogr., 1965, 79, 1-21)

67 M. F. Norman. A probabilistic model for free-responding. December 14, 1964.

68 W. K. Estes and H. A. Taylor. Visual detection in relation to display size and redundancy of critical elements. January 25, 1965, Revised 7-1-65. (Perception and Psychophysics, 1966, 1, 9-16)

69 P. Suppes and J. Donio. Foundations of stimulus-sampling theory for continuous-time processes. February 9, 1965. (J. math. Psychol., 1967, 4, 202-225)

70 R. C. Atkinson and R. A. Kinchla. A learning model for forced-choice detection experiments. February 10, 1965. (Br. J. math stat. Psychol., 1965, 18, 184-206)

71 E. J. Crothers. Presentation orders for items from different categories. March 10, 1965.

72 P. Suppes, G. Groen, and M. Schlag-Rey. Some models for response latency in paired-associates learning. May 5, 1965. (J. math. Psychol., 1966, 3, 99-128)

73 M. V. Levine. The generalization function in the probability learning experiment. June 3, 1965.

74 D. Hansen and T. S. Rodgers. An exploration of psycholinguistic units in initial reading. July 6, 1965.

75 S. C. Arnold. A correlated urn-scheme for a continuum of responses. July 20, 1965.

76 C. Izawa and W. K. Estes. Reinforcement-test sequences in paired-associate learning. August 1, 1965. (Psychol. Reports, 1966, 18, 879-919)

77 S. L. Blehart. Pattern discrimination learning with Rhesus monkeys. September 1, 1965. (Psychol. Reports, 1966, 19, 311-324)

78 J. L. Phillips and R. C. Atkinson. The effects of display size on short-term memory. August 31, 1965.

79 R. C. Atkinson and R. M. Shiffrin. Mathematical models for memory and learning. September 20, 1965.

80 P. Suppes. The psychological foundations of mathematics. October 25, 1965. (Colloques internationaux du Centre National de la Recherche Scientifique. Editions du Centre National de la Recherche Scientifique. Paris: 1967. Pp. 213-242)

81 P. Suppes. Computer-assisted instruction in the schools: potentialities, problems, prospects. October 29, 1965.

82 R. A. Kinchla, J. Townsend, J. Yellott, Jr., and R. C. Atkinson. Influence of correlated visual cues on auditory signal detection. November 2, 1965. (Perception and Psychophysics, 1966, 1, 67-73)

83 P. Suppes, M. Jerman, and G. Groen. Arithmetic drills and review on a computer-based teletype. November 5, 1965. (Arithmetic Teacher, April 1966, 303-309.

84 P. Suppes and L. Hyman. Concept learning with non-verbal geometrical stimuli. November 15, 1965.

85 P. Holland. A variation on the minimum chi-square test. (J. math. Psychol., 1967, 3, 377-413).

86 P. Suppes. Accelerated program in elementary-school mathematics -- the second year. November 22, 1965. (Psychology in the Schools, 1966, 3, 294-307)

87 P. Lorenzen and F. Binford. Logic as a dialogical game. November 29, 1965.

88 L. Keller, W. J. Thomson, J. R. Tweedy, and R. C. Atkinson. The effects of reinforcement interval on the acquisition of paired-associate responses. December 10, 1965. (J. exp. Psychol., 1967, 73, 268-277)

89 J. I. Yellott, Jr. Some effects on noncontingent success in human probability learning. December 15, 1965.

90 P. Suppes and G. Groen. Some counting models for first-grade performance data on simple addition facts. January 14, 1966. (In J. M. Scandura (Ed.), Research in Mathematics Education. Washington, D. C.: NCTM, 1967. Pp. 35-43.

91 P. Suppes. Information processing and choice behavior. January 31, 1966.

92 G. Groen and R. C. Atkinson. Models for optimizing the learning process. February 11, 1966. (Psychol. Bulletin, 1966, 66, 309-320)

93 R. C. Atkinson and D. Hansen. Computer-assisted instruction in initial reading: Stanford project. March 17, 1966. (Reading Research Quarterly, 1966, 2, 5-25)

94 P. Suppes. Probabilistic inference and the concept of total evidence. March 23, 1966. (In J. Hintikka and P. Suppes (Eds.), Aspects of Inductive Logic. Amsterdam: North-Holland Publishing Co., 1966. Pp. 49-65.

95 P. Suppes. The axiomatic method in high-school mathematics. April 12, 1966. (The Role of Axiomatics and Problem Solving in Mathematics. The Conference Board of the Mathematical Sciences, Washington, D. C. Ginn and Co., 1966. Pp. 69-76.

(Continued on inside back cover)

STUDENT PERFORMANCE IN COMPUTER-ASSISTED INSTRUCTION IN PROGRAMMING

by

J. E. Friend, J. D. Fletcher, and R. C. Atkinson

TECHNICAL REPORT NO. 184

May 10, 1972

PSYCHOLOGY & EDUCATION SERIES

This research was sponsored by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract No. N00014-67-A-0112-0054, Contract Authority Identification Number, NR No. 154-326.

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

STANFORD UNIVERSITY

STANFORD, CALIFORNIA

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Institute for Mathematical Studies in the Social Sciences Stanford University	Unclassified
	2b. GROUP

3. REPORT TITLE

Student Performance in Computer-assisted Instruction in Programming

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Technical Report

5. AUTHOR(S) (First name, middle initial, last name)

Jamesine E. Friend, John D. Fletcher, and Richard C. Atkinson

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
10 May 1972	90	7

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-67-A-0012-0054	Technical Report No. 184
b. PROJECT NO. NR 154-326	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Personnel and Training Rsearch Programs, Office of Naval Research Arlington, Va. 22217

13. ABSTRACT

An instructional system for teaching Algebraic Interpretive Dialogue (AID) to college-age students, two control programs (one for presenting instructional material and one for interpreting students' AID productions), and data collected by the two control programs are described. The first 21 lessons of the course and classification of the lesson exercises are also described. Data based on student daily reports are presented and discussed. Item analyses of data gathered by the instructional program, including stepwise linear regression models of item difficulty and analyses of selected data collected by the interpreting program are presented and discussed.

The following were among the results of this investigation: Although first response errors were often those of AID syntax, these errors were easily corrected in subsequent responses, and, in general, the syntax of AID commands was easily mastered; although students easily learned the mechanics of the instructional system, they rarely used features that allowed student control of instructional content and sequence; algebraic formulation of problems appeared to be more difficult than transforming algebraic expressions into AID commands; students had the greatest difficulty understanding hierarchy of arithmetic operations, use of functions, and the execution sequence of AID commands.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Algebraic Interpretive Dialogue (AID) Computer-assisted Instruction (CAI) Computer Programming Computer Science Education Johnniac Open-shop System (JOSS) Programming Languages						

STUDENT PERFORMANCE IN COMPUTER-ASSISTED INSTRUCTION IN PROGRAMMING

J. E. Friend, J. D. Fletcher, and R. C. Atkinson

Stanford University

Stanford, California 94305

1. Introduction

In 1967 the Institute for Mathematical Studies in the Social Sciences received a three-year grant* from the National Aeronautics and Space Administration to do exploratory research in the optimization of instruction and to develop a practical course of study using computer-assisted instruction (CAI). The course that was developed was a one-quarter college course in computer science (Friend & Atkinson, 1971). Developmental testing of the course was accomplished using NASA personnel and Stanford students. The course was not the only product of this grant; an entire computer-assisted instructional system (Friend, 1971) was developed, with this course as the first application.

The following year, the National Science Foundation provided funds** for a second, and very similar, application of the instructional system developed under the NASA funding. The second course was an introduction to computer programming for culturally-deprived high school students. This course taught the programming language, BASIC, whereas the first course taught AID. Both AID and BASIC are higher-order algebraic languages analogous to ALGOL and FORTRAN. The main difference between the

* NASA Grant NGR-05-020-244
**NSF Grant No. GJ-443X

two courses is that the AID course was written for college students with a good background in algebra, and the BASIC course was written for high school students with low reading ability and little or no background in algebra. The BASIC course was adequately tested with a group of over 100 students in an inner-city high school in San Francisco.

In 1970, the Office of Naval Research awarded a research contract to the Institute to continue its program of basic reseach in instructional strategies using the already developed instructional system and courses as research tools. The main subject of study was to be the AID course, but data collected in connection with the use of the BASIC course was also to be used if appropriate.

This report is a preliminary discussion of the research conducted under the ONR contract and is concerned only with the AID course.

2. The Instructional System

The course "Computer-assisted Instruction in Programming: AID" is an introductory course in computer science for community or junior college students with some background in high-school algebra. The course is completely self-contained and requires no supervision from a qualified instructor of programming. A brief student manual is supplied to supplement the instruction given by computer.

The computer used is a Digital Equipment Corporation PDP-10 located at Stanford University and owned and operated by the Institute for Mathmatical Studies in the Social Sciences. Connected to this computer by telephone lines are "Model-33" teletypewriters located in the schools and used for communicating with students. Instructions are printed on the teletypewriter terminal, and the student responds by typing his

replies on the same terminal. Teletypewriter operation is simple and can be learned from short instructions printed in the student manual.

Once the student has the teletypewriter in operation, all further instruction is given by computer under the control of a program known as INST. This program, which is the major component of the INSTRUCT system, interprets coded lessons providing individualized, tutorial instruction to the student. This instructional system and the method of programming lessons for it are described adequately by Friend (1971), and a detailed description will not be repeated here.

The AID course uses most of the features of the INSTRUCT system. The course contains 50 lessons organized into seven "lesson blocks." Each lesson block contains five tutorial lessons, followed by a self-test and a general review. The 50th lesson is a concluding lesson independent of the lesson blocks. The structure of the main strand is shown in Figure 1. The lessons vary in length from 10 to 60 exercises depending upon the content. Lessons of average length require about one hour to complete. Lesson length is completely under student control, and a student may take a few exercises or several lessons in one sitting.

One of the primary teaching strategies used in the course is the provision for student control of the sequence of instruction. Students may skip from any exercise in the course to any other exercise at any time, retracing their steps if they wish, or skipping lessons entirely. This strategy is intended to encourage the student to take responsibility for learning the concepts, not simply for progressing through a given set of exercises. Most college students are capable, and desirous, of assuming this responsibility, and the provision for student control of

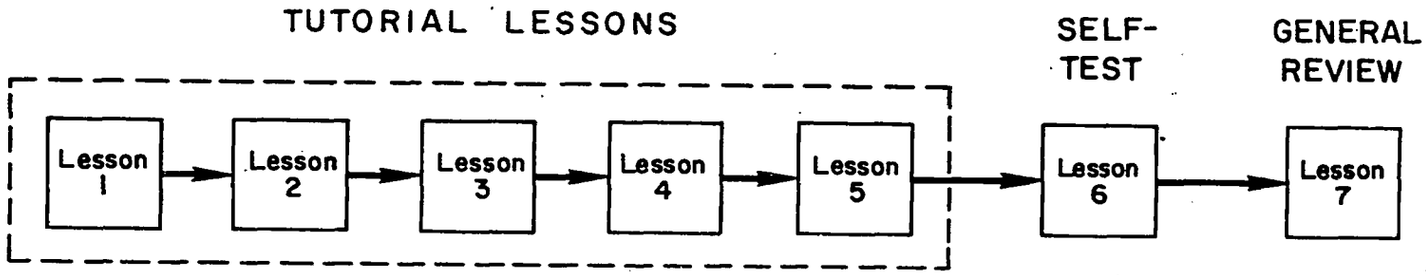

Figure 1. Structure of main lesson strand.

instruction is assumed to provide motivation. Whether or not all students are motivated by this treatment and whether or not it is an effective strategy in terms of the amount of learning taking place remains to be tested.

Because of this allowance for student control, the 50 lessons may be taken in any sequence. If the student does not exercise his prerogative for choosing the sequence, the lessons are automatically sequenced for him; and it is assumed that most students will, in fact, do the lessons in the order indicated.

Besides the main strand of lessons, the course also contains review lessons, one for each of the tutorial lessons in the seven lesson blocks. These review lessons are also tutorial and cover the same concepts as do the lessons they are associated with. However, they present each concept from a slightly different viewpoint providing additional practice in the skills to be learned. In general, each lesson covers five or six related concepts. In review lessons, the student may review whichever concepts he wishes, in any order he chooses. In fact, he must choose the order; there is no automatic sequencing provided by the program. At the end of each tutorial lesson, the student is asked if he wants to review any of the ideas covered in the lesson he has just completed. The student need not wait for these reminders, of course, since he can call for any review, or any exercise in any review, whenever he wishes.

Also associated with each tutorial lesson is a summary of the lesson, and the student is reminded at the end of each lesson that summaries are available. Each summary is printed in an 8-1/2" × 11" form that can be torn off and saved by the student as a permanent record.

In addition to the main strand of lessons, the reviews, and the summaries, there is a strand of "extra-credit" problems containing more difficult programming problems to be solved by the more capable students. It is recommended to instructors that the solutions of these problems be submitted for extra-credit if the course is graded. Not every lesson has associated extra-credit problems because they are not always appropriate to the subject matter. If there are such problems, the student is asked if he wants to try them at the end of the lesson.

The relationship and sequence of lessons, summaries, reviews and extra-credit problems are illustrated in Figure 2.

As described above, the main strand of lessons is divided into seven-lesson blocks, five tutorial lessons followed by a self-test and a general review. Lessons numbered 6, 13, 20, 27, 34, 41, and 48 are self-tests. Lessons numbered 7, 14, 21, 28, 35, 42, and 49 are general reviews. The self-tests are optional and students are told at the beginning of each test that the test is for their information only and may be skipped. The tests cover the concepts taught in the preceding five lessons and provide a good indication of weaknesses and areas requiring review. The general review which follows the self-test is also optional and is recommended for students who do poorly in the self-test. The general review is programmed to call the reviews for individual lessons as subroutines. For example, the student may review parts of Lessons 1, 3, and 4 and skip the review of Lessons 2 and 5. If he wishes to review Lesson 2, he will take the same review he would have taken if he had chosen to review the lesson immediately after taking it, and he can decide which concepts from that lesson to review and in which order.

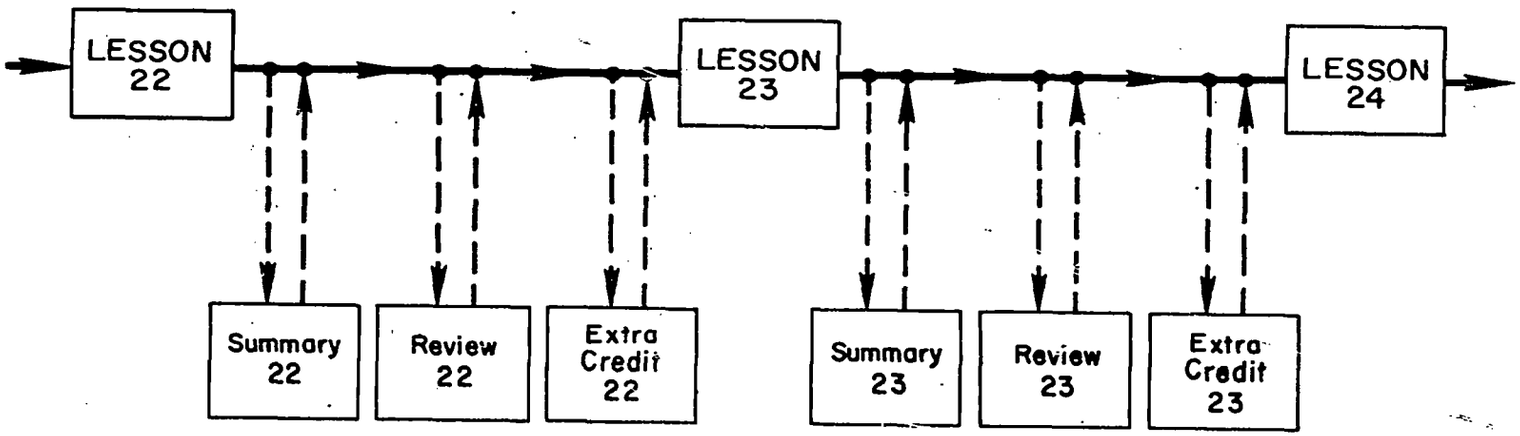

Figure 2. Relationship and sequence of tutorial lessons, summaries, reviews and extra-credit problems.

This review system is a gross method for providing individualized remediation. A more sensitive means of individualizing remediation is used within the lessons themselves where remedial sequences of exercises are given immediately to students who demonstrate an inadequate understanding of the material being taught. The remedial sequences that are imbedded within the lessons are not optional and are automatically provided for students whose responses indicate a lack of understanding. Because of this automatic remediation, different students may receive different numbers of exercises in a given lesson.

A student who makes an incorrect response to an exercise may not need an entire sequence of remedial exercises. He may profit from a single specific corrective message, pointing out the error and allowing him another try at the same problem. This kind of specific correction is used for most exercises in the course. Messages are provided, not for all possible incorrect responses, but for those incorrect responses judged to be most likely to occur.

In line with the provision for student control of the sequence of instruction, there is also provision for student control of the amount of instruction. This is done by providing additional instruction for almost every exercise in the form of "hints" and sample correct answers. After the problem statement is typed, the student is free to request a hint by typing a question mark. For many problems, a sequence of three or four hints is provided. The student may request one hint, make an attempt to answer the exercise, then ask for a second hint, and so on. Also, the student may ask for the correct answer at any time, either before or after he makes a try at the problem. He does this by typing

two keys simultaneously, the CTRL (control) key and the letter T. The answer will be printed, and he will proceed to the next exercise.

In the tutorial lessons, there is no limit on the number of attempts that a student may make for an exercise. As soon as he makes a correct response he will be given the next exercise in sequence, but if he fails to respond correctly, he is given another chance. If he cannot do the problem at all, he can judge for himself whether to continue trying or whether to go on with the lesson; whenever he decides to proceed without giving a correct response he can do so by typing CTRL-T, thereby getting the correct answer and the next exercise.

The CTRL key is also used in several other student controls. CTRL-G is used when the student wants to jump to a different exercise; after he types CTRL-G, the computer will ask "Where to?" and he can type the lesson and problem number that he desires. CTRL-U is used to erase entire lines, and CTRL-Z is used to stop the instructional program and end the session.

3. The AID Interpreter

The AID language was chosen as the subject of the course, not only because it is a useful algebraic programming language, but also because it is easily learned by a beginner. AID was developed, under the name of JOSS (Mark & Amerding, 1967; Shaw, 1965), by the Rand Corporation for use by scientists and engineers at Rand who needed an easily learned programming language that was capable of performing complex algebraic tasks. JOSS was later implemented for several other computers under a variety of names.

One of the advantages of AID as a beginner's language is that it is an interpreted, rather than compiled, language, which will act immediately on direct commands given by the user. Because of this feature the syntax and use of many AID commands can be taught to the student before he is taught about stored programs. In the AID course, the use of AID is introduced in Lesson 2 and the concept of stored programs is not introduced until Lesson 10.

A second advantage of AID, as compared to, say, FORTRAN or LISP, is that the syntax is a subset of English; commands are easily read as English imperative sentences. As an example, here is a simple AID program:

```
1.1  SET X = 1.3074.
1.2  SET Y = X/37.5.
1.3  TYPE Y IF Y < .029.
1.4  TYPE Y - .01 IF Y > .029.
1.5  TYPE "EUREKA" IF Y = .029.
```

As with most programming languages, AID is easier to read than to write, and considerable practice is needed by most students before they can produce a complete simple program like the above without error. However, they can begin to produce simple direct commands like

```
SET Y = X/37.5
```

or

```
TYPE Y - .01
```

on their first day.

Though simple to learn, AID is nevertheless a powerful tool for algebraic tasks. Definitions of conditional functions and recursive functions are relatively simple, and there are a variety of useful

standard functions, such as the basic trigonometric, exponential, and logarithmic functions. Lists and matrices can be easily defined, although AID does not provide the standard matrix manipulation functions found in some other programming languages.

Basic programming features, common to all programming languages, are available in AID. Variables and functions can be labeled, as can stored commands. There are branching commands, subroutine calls, and conditional clauses, and there are input and output commands for both disk and teletype.

AID commands and programs are interpreted by a program called, aptly, the AID interpreter. The interpreter used by the students of this course is a program written by Digital Equipment Corporation, the manufacturer of the Institute's PDP-10 computer.

The use of the AID interpreter is taught in the course, and the students are expected to use it frequently to solve problems given them in the lessons. Thus, the students taking this course will use two programs: INST, the instructional program which talks about the AID language, and AID, the interpreter which uses the AID language. In a previous version of the course, the two programs were completely independent and the students were required to learn to start and stop both programs so that they could switch back and forth to do the programming problems. This method was feasible but awkward and time consuming. For the present version of the course, both programs were modified to permit easy access from the instructional program to the AID interpreter. Students can call the AID interpreter at any time simply by typing the word "AID," and can return to the instructional program by typing "INST." This interface provides an additional advantage for research in that it supplies an

easy way to cross-reference data collected by the two programs. It is now possible to pass information, invisibly to the student, from one program to the other; every time the student calls the AID interpreter, the student's identity and current position in the course are passed to the AID interpreter so that data collected by AID can be keyed to data collected by INST.

4. Data

Both the instructional program and the AID interpreter contain data-collection subroutines that enable them to store information about student responses as the student is working. These data are stored temporarily on the disk and later transferred to permanent tape storage.

The instructional program records the following information with each student response.:

1. Student number. (Each student is assigned a unique number when he first enrolls for any computer-assisted instruction offered by the Institute.)
2. Date.
3. Time of day.
4. Lesson identifier.
5. Problem number.
6. Subproblem number.
7. Trial number.
8. Student response. (This is an exact character-by-character record of the response made by the student, excluding erasures made by the student.)

9. Analysis of correctness. (This is a record of how the instructional program scored the student's response to this exercise.)
10. Lesson score. (This is a cumulative score of the student's first responses to exercises within this lesson.)
11. Number of hints. (This is a record of the number of hints requested by the student before he made this reponse.)
12. Answer provided by program? (This switch records whether or not the student response was a request for the correct answer.)

It is estimated that about 3,000 such blocks of individual response data are collected for each student taking the course.

The AID interpreter also collects data but the form is simpler since it contains no routines to analyze the student input. The AID interpreter collects the following information:

1. Student number.
2. Date.
3. Lesson identifier. (This is information sent to the AID interpreter by the instructional program.)
4. Problem number. (This also is sent by the instructional program.)
5. Subproblem number. (Again, this is sent by the instructional program.)
6. Student input. (This is an exact character-by-character duplicate of everything typed by the student, excluding erasures.)

It is estimated that about 300 such blocks of data are collected for each student taking the course.

A variety of students have been enrolled in the AID course since the data collection routines were added to the programs. One of the largest

groups is comprised of students in the "High Potential" program at the University of California at Los Angeles. These students are entering freshmen who do not meet the usual entrance requirements for UCLA but who, for one reason or another, are suspected to have a higher potential than revealed by their high school records or by entrance examinations. These students may be described as "culturally deprived," and are products of inner-city schools where average achievement is quite low.

A second large group of students are from DeAnza College in Cupertino, California. DeAnza is a community college located a short distance from Stanford. The DeAnza students who have enrolled in the course were all unprovisionally admitted to DeAnza and most of them have a better high school background than the UCLA students.

A number of NASA personnel, from the Ames Research Center at Moffett Field, California, and from the Manned Spacecraft Center in Houston, Texas, have also enrolled for the course.

The Institute is providing CAI for hearing impaired students in several schools, and a half-dozen of these handicapped youngsters have also enrolled for the course during the last year.

5. The Curriculum: Lessons 1 to 21

The content of the course "Computer-assisted Instruction in Programming: AID" has been described elsewhere (Friend & Atkinson, 1971), so a complete description of the entire course will not be repeated here. The research reported in this paper concerns only the main strand Lessons 1 to 21, and these lessons are described below in considerable detail.

Lessons 1 to 21 contain three "blocks" of lessons; each of the blocks contains five tutorial lessons, one self-test and one general review:

Lessons 1 to 5 - Tutorial

Lesson 6 - Self-test of Lessons 1 to 5

Lesson 7 - General Review of Lessons 1 to 5

Lessons 8 to 12 - Tutorial

Lesson 13 - Self-test of Lessons 8 to 12

Lesson 14 - General Review of Lessons 8 to 12

Lessons 15 to 19 - Tutorial

Lesson 20 - Self-test of Lessons 15 to 19

Lesson 21 - General Review of Lessons 15 to 19

These lessons constitute a brief introduction to programming, covering such concepts as stored programs, use of variables, fundamentals of input and output, the syntax of algebraic expressions and Boolean statements, definitions of functions, conditional clauses and branching, core and disk storage, use of subroutines, and some debugging techniques. Some of these concepts (input and output, core and disk storage, subroutines) are discussed very briefly, while others (syntax of algebraic expression, syntax and meaning of Boolean statements) are covered more extensively. One major programming essential that is not introduced in the first 21 lessons is the loop, which is introduced in Lesson 23. Lists, arrays, trigonometric and exponential functions, recursive functions and the truth function are also introduced in later lessons. A brief outline of Lessons 1 to 21 is given in Table 1.

The 15 tutorial lessons in the first 21 lessons vary in length, depending upon the concepts covered by the lesson. Lesson 12, for example, covers two very simple commands and contains only 14 exercises, whereas Lesson 15 introduces Boolean statements and conditional clauses and contains 62 exercises. Also, the kinds of exercises in the lessons vary

Table 1

Brief Outline: Lessons 1 to 21

Lesson 1	Using the Instructional Program
Lesson 2	Using AID for Arithmetic
Lesson 3	Order of Arithmetic Operations
Lesson 4	Exponents and Scientific Notation
Lesson 5	The SET and DELETE Commands
Lesson 6	Test of Lessons 1 to 5
Lesson 7	General Review of Lessons 1 to 5
Lesson 8	The LET Command
Lesson 9	Some Standard AID Functions
Lesson 10	Indirect Steps, the DO Command, the FOR Clause
Lesson 11	Parts
Lesson 12	The DEMAND Command
Lesson 13	Test of Lessons 8 to 12
Lesson 14	General Review of Lessons 8 to 12
Lesson 15	Relations and the Use of the IF Clause
Lesson 16	The TO Command
Lesson 17	Debugging Techniques
Lesson 18	The Indirect Use of the DO Command
Lesson 19	Debugging, Permanent Storage
Lesson 20	Test of Lessons 15 to 19
Lesson 21	General Review of Lessons 15 to 19

with the subject matter. Lesson 15 has 24 true-false exercises, reflecting the content of the lesson (Boolean statements). Lesson 8, in contrast, has no true-false exercises, instead it has 32 exercises that require the student to predict the result of using given AID commands. Table 2 shows the number and type of exercises in Lessons 1 to 21, excluding the general review Lessons 7, 14 and 21.

The exercises are categorized into the 13 different problem types listed in Table 2. The first four types are multiple-choice exercises. In the AID course, multiple-choice exercises may have more than one correct choice, and the student response is not correct unless all correct choices are listed. The multiple-choice exercises in each lesson are all classified according to the number of correct choices except for the few that include the choice,

N. NONE OF THE ABOVE,

and are classified separately. Of the 675 exercises in the first 21 lessons there are 80 multiple-choice exercises, 56 of which have a single correct choice.

There are a number of exercises that appear to be constructed-response exercises in that the student is not presented with a list of possible answers from which to choose. However, closer inspection reveals that there are actually a limited number of choices of a form clearly implied in the statement of the problem. The following exercises, for example, imply only two choices:

Lesson 19, Exercise 8.

> SUPPOSE AID FOUND A SYNTAX ERROR IN STEP 17.2. DO YOU HAVE TO DELETE STEP 17.2 BEFORE YOU RETYPE IT?

Table 2

Number of Exercises, by Type, in Lessons 1 to 21
(Excluding General Reviews: Lessons 7, 14, 21)

Lesson Number	1	2	3	4	5	6	8	9	10	11	12	13	15	16	17	18	19	20	Total
Multiple-Choice 1 Correct Choice	3	4	7	6	5	2	1	1	2			2	5	5	1	6	1	5	56
Multiple-Choice 2 Correct Choices		4	1		1	2			1	1		1						1	12
Multiple-Choice 3 or More Correct Choices	2	2	2			1			1										8
Multiple-Choice Correct Choice: NONE		1	1	1		1													4
Total Multiple-Choice Exercises	5	11	11	7	6	6	1	1	4	1	0	3	5	5	1	6	1	6	80
Yes-No Exercises (except opinion questions)	2													1			7	1	11
True-False						9							24					1	34
Other Implied-Choice		1		1									2				1		5
Total Implied Choice	2	1	0	1	0	9	0	0	0	0	0	0	26	1	0	0	8	2	50
Predicted AID Response		4	19	25	5	12	32	17	1			13	6					1	135
Constructed AID Command		2	7	1	9	5	8		2	6	1	8		1			7	6	63
Reported Result of AID Use		3	3	3	13	2	4	6	4	4	3	1	1						47
Other Constructed-Response Exercises	3		1	15	17	4	8		13	5	2	3	10	10	18	22	4	10	145
Total Constructed Response	3	9	30	44	44	23	52	23	20	15	6	25	17	11	18	22	11	17	390
"Use AID"		3	3	3	9	2	4	3	5	6	5	2	6	2		1	4		58
Opinion (or preference)	5	4	5	6	4	3	5	4	4	8	3	3	8	8	4	6	8	2	90
Unclassified Exercises	3														4				7
Total	18	28	49	61	63	43	62	31	33	30	14	33	62	27	27	35	32	27	675

Lesson 6, Exercise 8.

ANSWER TRUE OR FALSE:
CTRL-Z WILL STOP THE AID INTERPRETER.

Lesson 4, Exercise 27.

IF YOU USED THIS COMMAND
TYPE 1/100
WOULD AID GIVE THE ANSWER IN DECIMAL FORM OR IN
SCIENTIFIC NOTATION?

Exercises of this type are labeled "implied choice" exercises and are classified as yes-no exercises, true-false exercises, or "other implied-choice" exercises. In the first 21 lessons there are 50 such exercises.

Three-hundred ninety-two of the exercises in the Table 2 lessons fall into the third major group comprising true constructed-response exercises. This group is subdivided into four classes: predicted AID responses, constructed AID commands, reported result of AID use, and "other constructed-response" exercises. The predicted AID responses contain questions like "What would AID answer if you gave this command ....?" One-hundred and five of the constructed-response exercises are of this type.

A smaller class of constructed response exercises includes the 63 exercises that require the student to construct a complete AID command. This class does not include all, or even most, of the AID commands that the student must construct. It includes exercises that pass the constructed AID commands to the instructional program; excluded are exercises that require direct communication with the AID interpreter. The instructional program is more capable than the AID interpreter of analyzing

constructed AID commands in detail and of giving meaningful messages to students who make errors when they first learn a new command. The usual sequence of instruction in introducing a new AID command is as follows:

First, an example of the command is shown, and its use is explained. In this step, students are usually required to determine which of several forms of the command are syntactically correct.

Second, the student is shown several examples and asked to determine what AID would respond if such a command were given to the AID interpreter.

Third, the student is asked to construct commands that would result in a specified action.

Fourth, the student is asked to start the AID interpreter and give commands of the new kind directly to AID.

Fifth, after using the AID interpreter as directed, the student is requested to report on the results given him by AID.

Exercises in this fifth step are classified in Table 2 as "reported result of AID use." In these exercises, the instructional program can infer the kind of errors a student is making and give him remedial instruction if needed. The device of asking students to switch to the interpreter for computation and to switch back to the teaching program to report results is one of the weakest features of the course, and it is used only from necessity. Clearly, an efficient means of interfacing the two programs is needed to pass information invisibly, and provide meaningful instruction for the students who need it. Only 47 of the 390 constructed-response exercises are of the type needed in the fifth step, because an overall teaching strategy used in the course is to

encourage students to assume responsibility for ferreting out their own errors and for determining if they have a correct program.

The fourth class of constructed-response exercises, called "other constructed-response exercises," contains 145 exercises of considerable variety. This class will be further subdivided in a later analysis.

The miscellaneous exercises contain a class designated in Table 2 as "use AID." These exercises require the student to use the AID interpreter in solving a problem. The problems range in complexity from copying commands in order to observe their consequences to solving complex problems by writing and debugging complete programs. Many of these exercises contain three or four problems, and the student is asked to solve all of the stated problems before he switches back to the instructional program and reports the results.

Also in the miscellaneous group are those exercises that elicit an opinion or preference. These exercises either ask the student to express a preference for the sequence of instruction (Do you want a summary of this lesson?) or to give a self-evaluation of his competence (Do you remember how to start the AID interpreter?). The exercises are fairly evenly distributed over lessons and there are 90 of them in the first 21 lessons.

In the following paragraphs, each of the first 21 lessons is described and characteristic exercises from the lesson are given.

Lesson 1: Using the Instructional Program

Lesson 1 is a set of 18 short exercises explaining how to use the instructional program. The mechanics of typing and erasing responses are explained, and instructions are given for starting and stopping the

program. The student is taught how to use optional control keys to get additional instruction (HINT and TELL commands) or to alter the sequence of exercises (the GO command).

The style of instruction in Lesson 1, as in succeeding lessons, is informal, and it does not always give explicit directions, requiring the student to attend the instructions carefully and learn by induction from the examples given.

Two of the exercises in Lesson 1 are given here.

Lesson 1, Exercise 3:

> IF MULTIPLE CHOICE PROBLEMS HAVE MORE THAN ONE CORRECT ANSWER, YOU CAN LIST THE CORRECT CHOICES IN ANY ORDER. SUPPOSE B, C AND D ARE THE CORRECT CHOICES FOR A PROBLEM. WHICH OF THESE WOULD BE CORRECT WAYS TO ANSWER?
>
> A. D, B, C, A
> B. B, D, C
> C. B, C, D
> D. D, B, C

Lesson 1, Exercise 14:

> FROM LESSON 1, YOU SHOULD HAVE LEARNED HOW TO SIGN ON AND OFF, HOW TO START AND STOP THE TEACHING PROGRAM, HOW TO GET A HINT, AND HOW TO USE CTRL-G. DO YOU WANT TO REVIEW ANY OF THESE TOPICS?

Exercise 14 illustrates an instructional strategy that is used in many lessons. At the end of a lesson, its content is briefly summarized, and the student is given the option of reviewing topics he is unsure of. If the student responds affirmatively, he is given the review associated with that lesson. In this way, the student is made responsible for the

material covered in each lesson and is forced to judge if he is competent to proceed with the course or if he needs additional instruction and practice.

In Lesson 1, five of the 18 exercises are multiple choice, similar in form to Exercise 3 shown above. This proportion of multiple-choice exercises is fairly typical of the course. The multiple-choice format is chosen over a constructed response format depending on the objective of the exercise. For example, if the purpose is to teach students to discriminate between syntactically correct and incorrect AID commands, a multiple-choice exercise is used; if the purpose is to teach the construction of an AID command, a constructed response will be requested.

Of the 13 exercises remaining in Lesson 1, seven could appropriately be labeled "implied multiple choice" since the statement of the exercise clearly implies only a small number of possible responses. In order to keep the terminology straight, however, exercises that are not in the conventional multiple-choice format, will be referred to as "implied choice" exercises, and the term, "multiple choice," will be reserved for exercises that list and label a set of choices and require that the student respond by typing the label or labels.

One of the most frequently used implied-choice exercises is the yes-no exercise. Many of these are designed to elicit opinion rather than information and have no "correct" answer. In Lesson 1, five of the seven implied-choice exercises are of this type and in Table 2 are classified as "opinion exercises" rather than "implied-choice exercises." This proportion is greater than that of later lessons, but still, indicates the style of the lessons.

Lesson 2: Using AID for Arithmetic

The 28 exercises in Lesson 2 teach the student how to start and stop the AID interpreter and how to use it for simple arithmetic by employing the "TYPE" command. The AID symbols for the four simple arithmetic operations (+, -, *, and /) are taught, and the use of parentheses in arithmetic expressions is introduced. By the end of the lesson, the student should be able to start the AID interpreter and give simple, direct commands such as:

```
TYPE  5/25
TYPE  3.25 + 17.4 - 3.12
TYPE  15 * 17 + 25 * 19
```

One of the most persistent errors made by students learning any algebraic programming language is the omission of the multiplication operator in algebraic expressions. The source of this difficulty is the convention of using juxtaposition to indicate multiplication. For example, we ordinarily write

$$a(b+c)$$

without an explicit multiplication operator, but AID, like other algebraic programming languages, demands that the multiplication be indicated explicitly. AID has an asterisk as the symbol for multiplication, as in

$$A * (B+C) .$$

In Lesson 2, there are a number of exercises aimed specifically at overcoming this error.

The following are exercises from Lesson 2.

Lesson 2, Exercise 15:

```
WHAT WOULD AID ANSWER TO THIS COMMAND?
TYPE 72/12
```

Lesson 2, Exercise 19:

USE AID TO DO THESE PROBLEMS:

1. FIND THE AREA OF A RECTANGLE WITH WIDTH 1.72375 AND LENGTH 12.001325.
2. SUPPOSE A SQUARE OF WIDTH .637825 IS CUT FROM THE ABOVE RECTANGLE. FIND THE AREA OF THE SQUARE.
3. FIND THE AREA OF THE REMAINING PART OF THE RECTANGLE.

Of the 28 exercises in Lesson 2, 11 are multiple choice, five are implied-multiple choice, and 12 are true constructed response exercises. The constructed response exercises vary in difficulty, from the simple problem given in Exercise 15, above, to the problem given in Exercise 19.

Lesson 3: Order of Arithmetic Operations

The arithmetic used in Lesson 2 was relatively simple, but in Lesson 3 the complexity increases with the addition of the concept of hierarchy of operations and the use of parentheses. Because each AID command must be typed entirely on a single line, horizontal division bars for grouping cannot be used. Thus, an expression like

$$\frac{xy}{z+3}$$

using more than one line of type, must be translated into the AID expression

$$X * Y / (Z+3)$$

with parentheses to show grouping. The AID expression is more difficult to construct, since it requires a conscious decision about the desired order of evaluation.

Lesson 3 teaches the student how to force an order of evaluation by using parentheses. To do this, the student must recognize the difference

between expressions like (16 - 4) - 3 and 16 - (4 - 3). After a few exercises on parentheses, rules are given for the hierarchy of the four basic arithmetic operators (+, -, *, and /), and a number of exercises are given in which the task is to determine the order of evaluation if no parentheses are used.

The following are two examples from Lesson 3.

Lesson 3, Exercise 5:

WHAT WILL AID ANSWER TO THIS COMMAND?
TYPE 1/(100/10)

Lesson 3, Exercise 10:

LOOK AT THESE THREE COMMANDS. AID WILL GIVE THE SAME ANSWER TO TWO OF THEM. WHICH TWO?

A. TYPE 3 + (2*4)
B. TYPE (3+2) * 4
C. TYPE 3 + 2 * 4

Since Lesson 3 is primarily a review of algebraic notions that may be better understood by some students than by others, it provides more opportunity for individualized branching. A good student can complete Lesson 3 in 27 exercises, but a poor student will be given additional practice and may do up to 49 exercises.

The exercises in Lesson 3 are of medium difficulty, and most can be done quickly. Eleven of the exercises are multiple choice, and an additional five are implied-choice exercises. Three of the exercises, including one with three parts, require the student to use the AID interpreter. Nineteen of the exercises are similar to Exercise 5 above.

Lesson 4: Exponents and Scientific Notation

Lesson 4 is longer than average and extends the work on arithmetic expressions to include exponentiation. First, the concept of exponentiation is reviewed, with the introduction of the AID symbol (↑). The rules for the hierarchy of operations are extended to include exponentiation, and the AID form of scientific notation is introduced. Negative exponents, fractional exponents, and zero as an exponent are also covered. Lesson 4, like Lesson 3, is largely review of algebraic principles and may have been forgotten. The exercises also provide practice in reading and constructing expressions in the form required by the AID interpreter.

Some examples from Lesson 4 follow.

Lesson 4, Exercise 6:

WHAT IS THE VALUE OF 5↑2/2?

Lesson 4, Exercise 12:

USE AID TO EVALUATE EACH OF THE FOLLOWING.

1. 4 SQUARED TIMES 3.1416
2. THE SUM OF 4 CUBED AND 6.
3. THE SUM OF THE SQUARES OF 1, 2, 3, 4, 5, 6, 7 AND 8

Lesson 4 contains 61 exercises, of which 48 require constructed responses. Most of the constructed responses require arithmetic calculations to answer questions like those illustrated in Exercise 6 above. Three exercises require the student to use the AID interpreter, and he is encouraged to use AID throughout the lesson.

Lesson 5: The SET and DELETE Commands

After the sizeable dose of arithmetic given in Lessons 3 and 4, Lesson 5 provides relief by returning to the mainstream of instruction

with the introduction of two new AID commands: the SET command and the DELETE command. SET is used in AID to assign values to real variables; DELETE is used to delete a previous assignment or definition. In AID, variables are single letters, and the SET and DELETE commands are easily learned by most students. A number of word problems are given to illustrate the use of the new commands. Lesson 5 also introduces the multiple-argument form of the TYPE command in which several TYPE commands can be combined into one by separating the arguments with commas (TYPE X,Y,X+Y).

The following exercises are from Lesson 5.

Lesson 5, Exercise 3:

WHAT WILL AID ANSWER AFTER THESE COMMANDS?

```
SET B = 1.5
TYPE 3*B
```

Lesson 5, Exercise 31:

TO FIND THE NEW AMOUNT IN A SAVINGS ACCOUNT, CALCULATE THE INTEREST AND ADD IT TO THE LAST BALANCE. START AID AND CALCULATE THE INTEREST AND THE NEW BALANCE AFTER ONE YEAR FOR AN ACCOUNT WITH AN INTEREST RATE OF 4.5 PERCENT PER YEAR AND A PREVIOUS BALANCE OF $3274.86. (ASK FOR A HINT IF YOU NEED ONE.)

WHAT IS THE INTEREST ON THE ABOVE ACCOUNT TO THE NEAREST PENNY?

WHAT IS THE NEW BALANCE IN THE ACCOUNT?

Lesson 5, with its 63 exercises, is fairly long and has nine exercises that ask the student to use AID to solve problems. The most difficult of these is shown in Exercise 31 above. The trend toward a

higher proportion of constructed responses continues here, with Lesson 5 having only six multiple-choice problems. There are four implied-choice exercises, but all of these are requests for opinions from the student.

Lesson 6: Test of Lessons 1 to 5

Lesson 5 concludes the first five-lesson tutorial block and is followed by a self-test in Lesson 6 and a general review in Lesson 7. Both Lessons 6 and 7 are optional. Lesson 6 contains 40 test questions and problems covering the material in Lessons 1 to 5. Like other self-tests, Lesson 6 supplies no hints, and students are allowed only one try on each exercise. However, the student may request the correct answers at any time by typing CTRL-T. Whenever a student misses an exercise, he is given a review reference and advised to review that topic before proceeding with the course.

The exercises in Lesson 6 are classified according to which lesson they are testing:

Lesson Number	Exercises in Lesson 6 Testing Given Lesson
1	2, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14, 15, 16
2	8, 11, 17, 18, 19, 20, 38-1, 39-1
3	21, 22, 23
4	24, 25, 26, 27
5	28, 29, 30, 31, 32, 33, 34, 35, 36, 37

Exercises 1, 1-1 and 40 are "opinion" exercises and are not listed. Exercises 38 and 39 are "use AID" exercises and are not listed. Of the 38 exercises in the list, 23 are constructed-response exercises.

Lesson 7: General Review of Lessons 1 to 5

The student is allowed to skip Lesson 7, but he is advised to take it if he missed more than five problems in the Lesson 6 self-test. Lesson 7 uses a more complex branching scheme to allow students to review only selected portions of the preceding lessons.

The following is an example from Lesson 7.

Lesson 7, Exercise 5:

> LESSON 4 WAS ABOUT EXPONENTS AND SCIENTIFIC NOTATION.
> FRACTIONAL EXPONENTS AND NEGATIVE EXPONENTS WERE
> DISCUSSED, AND ALSO THE USE OF 0 AND 1 AS EXPONENTS.
> THE ORDER OF ARITHMETIC OPERATIONS + - * / and % WAS
> COVERED.
>
> DO YOU WANT TO REVIEW ANY OF THESE THINGS?

If a student answers "yes," he is sent to the review lesson for Lesson 4, where he may review any of the lesson topics in any order he wants.

This first general review also reminds students that they can control the sequence of instruction by using the CTRL-G key and that the student manual provides an outline of the course.

Lesson 8: The LET Command

Lesson 8 is the beginning of a new lesson block, and introduces the powerful LET command, used in AID to define functions. The difference between LET and SET commands is discussed, and a variety of algebra problems is given. A major emphasis in Lesson 8 is on substituting arithmetic expressions for variables in algebraic expressions. Lesson 8 is the first lesson that has optional "extra-credit" problems.

The following are two exercises from Lesson 8.

Lesson 8, Exercise 4:

WHAT WILL AID ANSWER?

```
LET P(M) = M↑2
TYPE P(6/2)
```

Lesson 8, Exercise 28:

USE AID TO DO THIS PROBLEM. DEFINE A FUNCTION TO CONVERT DEGREES FAHRENHEIT TO DEGREES CENTIGRADE. THEN CONVERT THESE TEMPERATURES TO CENTIGRADE:

0, 10, 32, 72, 212

Of the 62 exercises in Lesson 8, over half are similar to Exercise 4 above. There are four multi-part problems that require the use of the AID interpreter. There is only one multiple-choice exercise in Lesson 8, and none of the exercises are more difficult than Exercise 28 above.

Lesson 9: Some Standard AID Functions

In Lesson 9, the student is introduced to four functions already defined in AID. These are:

SQRT(X) - the square root function,
IP(X) - the "integer part" function,
FP(X) - the "fraction part" functi...
SGN(X) - the sign function.

These functions, together with those defined by the student, are used in several problems requiring the use of the AID interpreter.

The following are two exercises from Lesson 9.

Lesson 9, Exercise 2:

WHAT WILL AID ANSWER?

```
TYPE 3 * SQRT(100)
```

Lesson 9, Exercise 14:

YOU CAN USE THE AID FUNCTION FP(X) TO FIND OUT IF ONE NUMBER CAN BE DIVIDED BY ANOTHER WITHOUT A REMAINDER....

IS 2976 EVENLY DIVISIBLE BY 3?

Lesson 9 has 31 exercises of which 17 are similar to Exercise 2 above.

Lesson 10: Indirect Steps, the DO Command, the FOR Clause

In Lesson 10, the concept of a stored program is introduced. Up to this point, students have been using AID as a desk calculator, doing all exercises with direct commands, i.e., commands that are executed immediately. In this lesson, the students are taught that TYPE commands can be stored for later execution by prefacing the command with a step number, as in the following examples:

```
2.1  TYPE F(16)
4.7  TYPE X↑2,X↑3 .
```

They are also taught how to execute these stored commands by using a DO command. Two variants of the FOR clause are used to modify DO commands. In the first variant, values for the iteration variable are given by a simple listing:

```
DO STEP 17.3 FOR Y = 1,2,7,14.3.
```

In the second variant, values for the iteration variable are given in a range specification that specifies an initial value for the variable, a step size, and a final value:

```
DO STEP 5 FOR X = 4(2)9.
```

This command specifies that X will assume the value 3, then be incremented by 2 after each iteration of step 5 until X > 9. This is equivalent to

the FORTRAN form,

DO 5 X = 4,9,2
5 < statement 5 >,

and the ALGOL form

FOR X ← 4 STEP 2 UNTIL 9 DO
< statement 5 >.

The following are some examples from Lesson 10.

Lesson 10, Exercise 12:

USING AID, WRITE AN INDIRECT STEP THAT WILL CONVERT MILES PER HOUR TO FEET PER SECOND. THEN CONVERT ALL OF THESE TO FEET PER SECOND:

10 MILES PER HOUR
100 MILES PER HOUR
65 MILES PER HOUR
1023 MILES PER HOUR

Lesson 10, Exercise 17:

WHAT VALUES OF A WILL BE USED IF THIS COMMAND IS GIVEN?

DO STEP 73.7 FOR A = 5(10)35

A. 5, 10, 15, 20, 25, 30, 35
B. 10, 15, 20, 25, 30, 35
C. 5, 15, 25, 35

Of the 33 exercises in Lesson 10, four are multiple choice and another four are implied choice. Most of the constructed-response exercises are quite simple, and Exercise 12 above is the only one that requires any problem-solving skills.

Lesson 11: Parts

Lesson 11 explains how indirect (stored) steps are grouped into "parts." Steps 12.1, 12.7, and 12.8, for example, are grouped as

"Part 12," and can be executed by a single command:

DO PART 12.

The sequence of execution depends only upon the numerical order of the step numbers, and not upon the sequence in which they were written. Thus, steps 3.75, 3.2 and 3.8 will be executed in the order: 3.2, 3.75, 3.8. This concept is clear to most students. The only difficulty is caused by step numbers with trailing zeros; some students fail to order correctly a sequence like 3.5, 3.8, 3.10 (the correct order is 3.10, 3.5, 3.8).

Here are two examples from Lesson 11.

Lesson 11, Exercise 5:

YOU CAN TYPE THE STEPS IN ANY ORDER, BUT AID WILL ALWAYS DO THEM IN NUMERICAL ORDER. WHICH STEP WILL BE DONE FIRST?

17.4 TYPE X↑Y
17.5 SET N=5
17.2 SET X=10
17.3 SET Y=2

Lesson 11, Exercise 11:

A PART (SET OF INDIRECT STEPS) IS ALSO CALLED A PROGRAM. USE AID TO WRITE A PROGRAM THAT WILL LIST THE RADIUS, DIAMETER, CIRCUMFERENCE, AND AREA OF A CIRCLE OF RADIUS R. THEN USE THE PROGRAM FOR R = 10, 20, 30, 40 AND 50.

Lesson 11 has 30 exercises, with an unusually high proportion of "opinion" questions (8 out of 30). Only one of the 30 exercises is multiple choice, and six of the 30 require the student to use the AID interpreter, a slightly higher proportion than was found in earlier lessons.

Lesson 12: The DEMAND Command

In Lesson 12 the DEMAND command is introduced. The DEMAND command is used in AID programs for teletype input. The DEMAND command, unlike previously introduced commands, can be used only indirectly, as a stored command. This lesson also introduces a variant of the DO command:

DO PART 7, 5 TIMES

The following are exercises from Lesson 12.

Lesson 12, Exercise 4:

START AID AND WRITE A PROGRAM THAT WILL ASK YOU FOR 3 NUMBERS, A, B, AND C, AND THEN GIVE YOU THE AVERAGE OF THE 3 NUMBERS. AFTER YOU HAVE TESTED YOUR PROGRAM, USE IT TO FIND THE AVERAGE OF

A = 179.053
B = 23.7
C = 271.0015

Lesson 12, Exercise 5:

WHAT COMMAND WOULD YOU USE IF YOU WANTED PART 2 DONE 7 TIMES?

Lesson 12 is very short, containing only 14 exercises. None is multiple choice, and five require the student to use the AID interpreter.

Lesson 13: Test of Lessons 8 to 12

Lessons 8 to 12 constitute the second five-lesson tutorial block of instruction and are followed by an optional self-test (Lesson 13) and a review (Lesson 14). Lesson 13 is structured like other self-tests; the student is given only one try for each exercise, and there are no hints provided. A student who cannot do an exercise can request the correct answer by typing CTRL-T.

The following classifies the exercises in Lesson 13 according to which lesson is being tested.

Lesson Number	Exercises in Lesson 13 Testing Given Lesson
8	1-1, 2, 3, 4, 5, 6, 7, 8
9	9, 10, 11, 12, 13, 14, 16
10	17, 18, 19, 20, 21, 22, 23, 24
11	25, 26, 27
12	28, 29-1

Five of the 33 exercises in Lesson 13 are not included in the list. Exercises 1, 15-1 and 30 are "opinion" questions, and Exercises 15 and 29 are "use AID" exercises. Of the 28 exercises in the list, 25 are constructed-response exercises.

Lesson 14: General Review of Lessons 8 to 12

Lesson 14, the general review of Lessons 8 to 12, is optional but is recommended for students who missed more than three problems in the preceding self-test. Here is one example from Lesson 14.

Lesson 14, Exercise 3:

> LESSON 8 WAS ABOUT THE "LET" COMMAND AND HOW TO USE IT TO DEFINE A FUNCTION. FUNCTIONS OF 2 AND 3 VARIABLES WERE DISCUSSED. INSTRUCTIONS FOR PRINTING AND DELETING A FUNCTION WERE GIVEN.
>
> DO YOU WANT TO REVIEW ANY OF LESSON 8?

The structure of Lesson 14, like that of other general reviews, allows a student who answers "yes" to branch to the review for the lesson and to review any of the lesson topics in any order.

Lesson 15: Relations and the Use of the IF Clause

Lesson 15 begins a new lesson block and introduces the most powerful programming tool: the conditional clause. The conditional (IF) clause may be appended to any of the commands so far introduced. The following AID symbols for arithmetic relations are introduced:

<	less than,
>	greater than,
<=	less than or equal,
>=	greater than or equal,
#	not equal.

The terms "positive," "negative," and "non-negative" are reviewed. The Boolean connectives "and" and "or" are also introduced. Students are required to write several programs using conditional branching.

The following are examples from Lesson 15.

Lesson 15, Exercise 14:

STUDY THIS PROGRAM.

49.5 TYPE X IF X > Y
49.6 TYPE Y IF X < = Y
DO PART 49.

IF X = 12.1 AND Y = 6, WHAT WILL AID ANSWER?

Lesson 15, Exercise 20:

WRITE A PROGRAM THAT WILL PRINT "SAME" IF ALL THREE NUMBERS X, Y, AND Z HAVE THE SAME SIGN. THE PROGRAM SHOULD PRINT "DIFFERENT" IF THE NUMBERS DO NOT ALL HAVE THE SAME SIGN.

Of the 62 exercises in Lesson 15, a large number (24) are true-false exercises. The students are required to use AID in six exercises similar to Exercise 20 above.

Lesson 16: The TO Command

Lesson 16 reviews the use of conditionals and introduces conditional branching.

The following are two examples from Lesson 16.

Lesson 16, Exercise 3:

HERE IS A PROGRAM THAT CALCULATES THE AREA OF A RECTANGLE OF LENGTH L AND WIDTH W. IF EITHER L OR W IS NEGATIVE, PART 15 IS USED TO GIVE AN "ERROR" MESSAGE.

```
14.1  DEMAND L
14.2  TO PART 15 IF L < 0
14.3  DEMAND W
14.4  TO PART 15 IF W < 0
14.5  TYPE L * W
15.1  TYPE "DO NOT USE NEGATIVE NUMBERS."
```

WHICH STEPS WILL BE DONE IF L = 5 AND W = - 3?

Lesson 16, Exercise 4:

WRITE A PROGRAM THAT WILL DEMAND A RADIUS R AND THEN CALCULATE THE AREA OF A CIRCLE WITH THAT RADIUS. USE TWO PARTS, ONE FOR THE MAIN PROGRAM AND ONE FOR AN "ERROR" ROUTINE TO BE USED IF R IS NEGATIVE.

There are 27 exercises in Lesson 16, with five multiple choice and eight "opinion" questions.

Lesson 17: Debugging Techniques

Lesson 17 concentrates on debugging techniques, showing the student how to trace a program by making a table listing the steps in order of execution.

Here is one example from Lesson 17, Exercise 3:

FOR PRACTICE, LET'S MAKE A TRACE OF THIS PROGRAM, ASSUMING A = 3.

```
31.3  DEMAND A
31.2  SET B = A↑2 - 10
31.3  SET C = A IF A > B
31.4  SET C = B IF A < = B
31.5  TYPE B
31.6  TYPE C
```

FILL IN THE VALUES OF C IN THIS TRACE (STARTING AT STEP 31.3).

STEP	A	B	C
31.1	3	-	-
31.2	3	-1	-
31.3	3	-1	?
31.4	3	-1	?
31.5	3	-1	?
31.6	3	-1	?

Of the 27 exercises in Lesson 17, 18 are similar to the exercise above. Four exercises require the student to write a complete trace with paper and pencil and check his answer by typing CTRL-T.

Lesson 18: The Indirect Use of the DO Command

In Lesson 18, the indirect use of the DO command is introduced. Up to this point, the student has been using DO commands directly to execute programs or single steps. The DO command can also be given indirectly to execute subroutines. A conditional clause is frequently used with indirect DO commands.

The following is an exercise from Lesson 18.

Lesson 18, Exercise 2:

WHEN AID COMES TO AN INDIRECT "DO" COMMAND, IT WILL DO THE STEP OR PART INDICATED AND THEN RETURN TO THE STEP AFTER THE "DO" COMMAND.

```
16.1  DO STEP 2.1 IF Q < 0
16.2  TYPE Q
 2.1  SET Q = - Q
DO PART 16
```

IF Q = 3, THE STEPS WILL BE DONE IN WHICH ORDER?

A	B	C	D
16.1	16.1	16.1	16.1
2.1	16.2	2.1	16.2
16.2	2.1		

There are 35 exercises in Lesson 18, with only one programming problem requiring the use of AID.

Lesson 19: Debugging, Permanent Storage

The first half of Lesson 19 is an optional section of tips for writing and debugging programs. This section is primarily for students who have been having difficulty with the programming problems in preceding lessons. The second half of the lesson describes the difference between core memory and disk storage and teaches the students how to store programs on the disk by using the AID file commands: USE, FILE, RECALL, and DISCARD.

Here are two examples from Lesson 19.

Lesson 19, Exercise 13:

WHAT COMMAND WOULD YOU USE TO FILE PART 29 AS ITEM 3?

Lesson 19, Exercise 17:

IF A NUMBER B WAS FILED AS ITEM 6, YOU WOULD RECALL IT BY TYPING

USE FILE 100

AND THEN WHAT?

Of the 32 exercises in Lesson 19, eight are requests for opinions, and four ask the student to use AID.

Lesson 20: Test of Lessons 15 to 19

Lesson 19 completes the third lesson block. Lesson 20 is a test of Lessons 15 to 19, and is structured like the tests in Lessons 6 and 13. Lesson 20 contains 27 exercises, of which two are requests for opinions. The other 25 can be grouped according to which lessons they test:

Lesson	Exercises in Lesson 20 Testing Given Lesson
15	1-1, 2, 3, 4, 5, 6
16	7, 16
17	8, 9, 10
18	11, 12, 13, 14, 15
19	17, 18, 19, 20, 21, 22, 23, 24, 25

There are six multiple-choice exercises, two "implied-choice" exercises, and 17 constructed responses.

Lesson 21: General Review of Lessons 15 to 19

Lesson 21 is a general review of the lessons tested by Lesson 20. Like other general reviews, it is optional and is recommended for students who missed more than three problems in the test.

Here is one example from Lesson 21:

Lesson 21, Exercise 8:

> LESSON 19 EXPLAINED HOW TO PLAN, WRITE, AND EDIT A PROGRAM; WHAT KINDS OF ERRORS THERE ARE AND HOW TO CORRECT THEM; AND HOW TO USE PERMANENT STORAGE.
>
> DO YOU WANT TO REVIEW LESSON 19?

A student who responds "yes" will be given the review lesson for Lesson 19.

6. The Daily Report

A daily report program was provided to inform teachers of the progress of individual students. The report lists the students in given classes, shows their current position in the curriculum, and indicates if they used the curriculum on the day of the report. The position of each student is indicated by printing the number of the last problem he completed in the tutorial lessons, in the reviews, in the summaries, and in the extra-credit problems. By comparison with previous daily reports, the teacher (or researcher) can judge about how much an individual student has progressed, and he can compare the positions of different students.

However, the daily report provides only a rough measure of progress since the students themselves control the sequence of instruction. A student may be on Lesson 5 one day and on Lesson 12 the next, either by diligently working through all the intervening lessons or by simply skipping directly to Lesson 12. Also, a student may decide to go back to review a previous lesson, and his daily report will show that he was on Lesson 12 one day and had regressed to Lesson 5 the next. Even if one assumes that students are working their way straight through the course, it is hard to get a precise measure of progress from the reports

because of the varying lengths of lessons and the uneven dispersion of time-consuming programming problems. Nevertheless, the daily reports can provide an adequate indication of average rate of progress through the course and of variation in the rates of progress among students.

To illustrate use of the daily reports, one class of 39 students was selected as a sample. All these students were enrolled in the UCLA "High Potential" program. The daily report information for the class is summarized for each student and presented in Table 3. Number of days worked, number of lessons completed, and rate of progress in number of lessons per day are listed in Table 3. This particular class was chosen because of its large enrollment, but it is atypical because attendance for AID was voluntary. Of 48 possible workdays, the average number of days worked by the 39 students is 10.4 with a range from 1 to 36 days. This average is high for strictly voluntary attendance. Average number of lessons completed is 14.1 with a range from 2 to 36 lessons, indicating that the students completed slightly less than one-third of the course in the time allotted.

To illustrate the individuality of the students, progress in number of lessons completed for three students (Nos. 2, 7, and 11) from the class tabulated in Table 3 is graphed in Figure 3. Student 2 can be characterized as slow and steady, Student 7 can be characterized as fast and steady, and Student 11 can be characterized as persistent but erratic.

7. Item Analysis of INST Data

In this preliminary report, only data from first responses for Lessons 1 to 21 are analyzed. The data used were collected by the INST

Table 3

Daily Report Summary for One Class of 39 Participating Students

Student	Number of days worked	Number of lessons completed	Number of lessons completed per day
1	22	23	1.04
2	17	11	.65
3	17	27	1.59
4	2	5	2.50
5	13	18	1.38
6	13	8	.62
7	15	24	1.60
8	17	26	1.53
9	1	9	9.00
10	1	2	2.00
11	36	36	1.00
12	6	4	.67
13	3	2	.67
14	2	4	2.00
15	6	5	.83
16	2	3	1.50
17	8	20	2.50
18	13	13	1.00
19	16	22	1.38
20	9	9	1.00
21	4	8	2.00
22	19	22	1.16
23	20	31	1.55
24	18	22	1.22
25	8	20	2.50
26	11	11	1.00
27	5	8	1.60

Table 3 (cont'd)

Student	Number of days worked	Number of lessons completed	Number of lessons completed per day
28	1	2	2.00
29	6	8	1.33
30	8	11	1.38
31	2	4	2.00
32	17	21	1.24
33	10	23	2.30
34	18	17	.94
35	7	15	2.14
36	6	8	1.33
37	20	20	1.00
38	1	7	7.00
39	5	21	4.20
Mean	10.38	14.10	1.86
S.D.	7.80	9.02	1.62
N	39	39	39

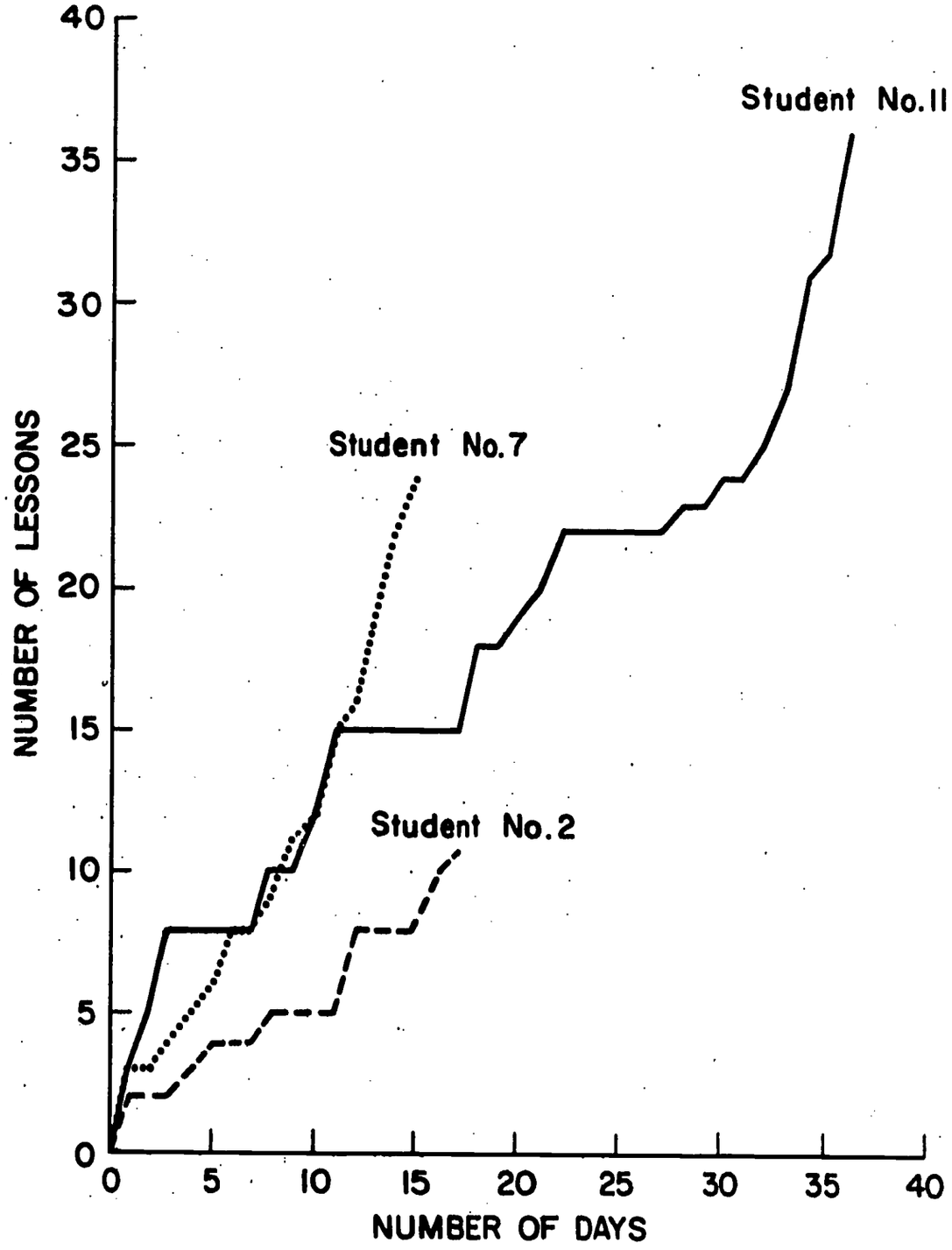

Figure 3. Student progress in number of lessons completed after each day of work in AID for three students.

instructional program. No data collected by the AID interpreter are included in this section.

Because of the unique branching structure used in the general review, Lessons 7, 14 and 21 are not included in the analysis. The self-test Lessons 6, 13 and 20 are included for comparison.

To describe the statistics reported, it is necessary to explicate the definitions of "first response" and of "correct response" used by the data collection and analysis routines. "First response" is defined to be the first of a set of contiguous responses made in any one session to a single exercise, with the following two exceptions. First, a student may, through his own volition or by automatic action of the program, repeat an exercise. In such a case, the first response made by the student the second time he encounters the exercise is counted again as a first response to that exercise. Thus a total of 35 first responses to a given exercise may be the work of only 34 students. Second, if a student terminates a session after responding incorrectly to an exercise, he will commence the next session with that same exercise; his first response in the subsequent session will be tallied as another first response to the exercise. In actual practice, the effect of these "extra" first responses is negligible.

Some responses are not considered at all in the tally of first responses. These are responses that cannot be classified as correct or incorrect. The only responses that fall into this category are question mark (student request for a hint), CTRL-G (student request for a change of problem sequence), CTRL-A (student request for a repeat of the problem),

and CTRL-Z (student request for sign-off). If any of these occur as a first response, they are ignored.

Excluding the lessons and exercises not graded, 6,512 first responses made by the 68 students who completed one or more AID lessons during the first semester of the 1972-73 school year were analyzed for this report. Of the 68 students, 39 were drawn from the UCLA "High Potential" program, 26 from Ames-NASA, and three from the network of schools for the deaf.

The other critical definition used is that of "correct response." In general, the definition of "correct" is supplied by the programming and is what one would expect. If the correct answer to an exercise is "true," the student response may be "true" or "t," and any other response such as "false" or "help" or "yes" is classified as incorrect.

Some exercises have no clearly defined correct answer, however. First, requests for sequencing, "Do you want to try the extra-credit problems for this lesson?", cannot be said to have a right or wrong answer. Second, the programming of a few exercises precluded easy classification; there were seven such exercises in the 675 exercises considered. Third, some exercises ask the student to use AID to solve a stated problem. We could consider the response to be correct if the student did indeed use AID, but on a less superficial level we need some analysis of the student's use of the AID interpreter. This implies a routine that can judge the correctness of a student-written program. Unfortunately, such a routine is not available. The general solution to the problem of proving the correctness of a computer program has been shown to be recursively unsolvable (Davis, 1958). While many particular cases of this problem are solvable, it remains a deep and non-trivial problem to construct an algorithm that will prove correctness for a comprehensive set of student

solutions. Some data collected by the AID interpreter have been hand-graded and will be described later in this report.

Problem Types

A summary of correct first responses by problem type is shown in Table 4. Number correct, number of first responses, and percent correct are shown for each problem type.

This summary reveals several interesting results from the item analysis. First, the proportion correct for all exercises is only 65.7%, a figure at least 10% lower than predicted after developmental testing of the program. This result is probably explained by the fact that most of the data for this analysis was obtained from students in a "High Potential" program comprising students who do not meet regular college entrance requirements but who are suspected to have higher ability than indicated by their achievement records.

In later analyses, the data used will be drawn from community college students who have been admitted unprovisionally. The original goal of the AID project was to prepare a curriculum for community college students, and these new data will provide a better assessment of the curriculum.

Second, an unexpected result is the proportion correct for multiple-choice exercises as compared with constructed responses. The proportion correct is markedly lower, 54.9%, for multiple-choice exercises compared with 66.6% for constructed responses. In presenting a lesson-by-lesson comparison of the proportion correct for multiple-choice and constructed responses Table 5 shows this result to be quite stable across the lessons.

Table 4

Number and Percent of Correct First Responses in Lessons 1 to 21
(Excluding Lessons 7, 14, 21)

	Total correct first responses	Total first responses	Percent correct
Multiple-Choice 1 Correct Choice	436	718	60.7
Multiple-Choice 2 Correct Choices	60	160	37.5
Multiple-Choice 3 or More Correct Choices	59	120	49.2
Multiple-Choice Correct Choice: NONE	23	55	41.8
Total Multiple-Choice Exercises	578	1053	54.9
Yes-No Exercises (except opinion questions)	99	133	74.4
True-False	215	254	84.6
Other Implied-Choice	47	59	79.7
Total Implied-Choice	361	446	80.9
Predicted AID Response	1282	1861	68.9
Constructed AID Command	497	798	62.3
Reported Result of AID Use	557	878	63.4
Other Constructed-Response Exercises	1003	1476	68.0
Total Constructed Response	3339	5013	66.6
Totals	4278	6512	65.7

Table 5.

Percent of Correct First Responses for All Multiple-choice and All Constructed-response Exercises in Lessons 1 to 21 (Excluding Lessons 7, 14, 21)

Lesson	Percent correct multiple-choice	Total responses multiple-choice	Percent correct constructed response	Total responses constructed response
1	66.7	81	68.0	50
2	67.3	153	58.2	146
3	32.7	101	65.1	373
4	57.4	61	71.5	368
5	46.7	75	70.2	601
6	44.7	76	69.5	298
8	30.8	13	67.2	862
9	26.3	19	70.1	472
10	76.3	76	81.8	357
11	70.6	17	73.6	208
12	--	0	77.8	99
13	54.3	35	58.1	296
15	61.2	85	55.3	237
16	47.6	84	65.2	178
17	66.7	12	0.0	55
18	48.1	81	50.6	77
19	25.0	12	67.4	135
20	61.1	72	51.7	201
Totals	54.9	1053	66.6	5013

Third, the proportion correct for implied-choice exercises (80.9%) is higher than either multiple-choice or constructed-response exercises. This result might be expected since all of the implied-choice exercises imply only two choices, and about 50% of the answers would be correct if they were selected by random guessing.

Fourth, the order of percent correct among the different kinds of multiple-choice exercises is not what would be predicted from the number of correct choices available. Exercises in which there are more than one correct choice are more difficult (119/280 = 42.5%) than are exercises in which exactly one choice is correct (459/773 = 59.4%). However, exercises in which none of the listed choices is correct are more difficult (41.8%) than the other single choice exercises (60.7%), and exercises in which there are three or more correct choices are easier (49.1%) than those in which there are two correct choices (37.5%).

A more detailed analysis of the data for multiple-choice exercises will be discussed later in this report.

Student Control of Amount of Instruction

Two features of the course that allow the student to control the amount of instruction he receives are "hints" and "tells." Most of the exercises have one or more optional hints which can be requested by the student at any time, either before or after making a response. Also, in all exercises for which there is a correct answer the student may request the correct answer, i.e., a tell, at any time.

How these two controls are used is as yet undetermined, and will be the subject of future research. The item analyses of first responses, on which this report is based, does provide some evidence, however. The

number of hints and the number of tells requested before first responses were tallied.

In some exercises, the hints provide information vital to the solution of the problem. This information was put into hint messages rather than into the problem text itself whenever it could be assumed that a fairly large proportion of the students would already know it. For example, if the exercise is to use AID in calculating the volume of a cylinder of given radius and height, it is essential to use the formula that gives volume in terms of radius and height. Many students can be assumed to know this formula so it was not included in the problem statement but instead was included as the first hint. In other cases, the hints suggest a strategy to be used in solving the problem. It was assumed that hints would be used freely by as many as a quarter of the students.

Table 6 shows the number of hints requested before the first response by problem type. Only 89 hints preceding first responses were requested for Lessons 1 to 21. Since there were over 6,500 first responses for these lessons, this number of requests for hints is surprisingly low. Variation between different lessons and different problem types should be viewed with skepticism because of the small number of requests.

It may be that students do not request hints until they have made at least one try, or they may believe they would be penalized for requesting hints, or they may not understand how to ask for them. Because the low number of hint requests is so unexpected it is worth pursuing in later research on use of control features.

Table 6

Number and Percent of Hints Requested in Lessons 1 to 21
(Excluding Lessons 7, 14, 21)

	Total hints requested	Total responses	Percent hints
Multiple-Choice 1 Correct Choice	4	718	0.6
Multiple-Choice 2 Correct Choices	0	160	0.0
Multiple-Choice 3 or More Correct Choices	0	120	0.0
Multiple-Choice Correct Choice: NONE	0	55	0.0
Total Multiple-Choice Exercises	4	1053	0.4
Yes-No Exercises (except opinion questions)	7	133	5.3
True-False	0	254	0.0
Other Implied-Choice	1	59	1.7
Total Implied-Choice	8	446	1.8
Predicted AID Response	30	1861	1.6
Constructed AID Command	19	798	2.4
Reported Result of AID Use	17	878	1.9
Other Constructed-Response Exercises	11	1476	0.7
Total Constructed Response	77	5013	1.5
Totals	89	6512	1.4

Requests for the correct answer were expected to be relatively low as a first response, about 5% as a first response and about 10% as a second, or later, response. Table 7 shows the number of first responses requesting the correct answer by problem type. Two hundred forty-two of the 6,512 first responses, or 3.7%, were requests for the answer. It is interesting that there was a greater proportion of such requests for multiple-choice exercises (4.6%) than for constructed-response exercises (3.8%), indicating that multiple-choice exercises are more difficult.

In the analysis of structural variables affecting problem difficulty, proportion correct is taken as the measure of difficulty. An interesting comparison could be made by using the number of requests for answers as a measure of difficulty. This was not done in the current analysis because the number of subjects was too small to permit meaningful application of the stepwise multiple linear regressions used below.

Lesson to Subtest Relationship

The percentages correct by lesson listed in Table 8 show little variation, with the exception of Lesson 17, and it can be inferred that the curriculum is reasonably consistent in difficulty. Lessons 10 and 12 appear to be the easiest, and Lessons 17 and 18 seem to be the most difficult.

Two of the more difficult lessons are tests (Lessons 13 and 20), and the average percent correct for the three tests (Lessons 6, 13, and 20) is 60.7%, which is slightly lower than the average for tutorial lessons, 63.3%. To compare tutorial lessons and tests, each test was divided into five subtests each consisting of the exercises associated with one of the five preceding tutorial lessons, and the percent correct

Table 7

Number and Percent of Requests for the Correct Answer (Tells) in Lessons 1 to 21 (Excluding Lessons 7, 14, 21)

	Total tells requested	Total responses	Percent tells
Multiple-Choice 1 Correct Choice	39	718	5.4
Multiple-Choice 2 Correct Choices	5	160	3.1
Multiple-Choice 3 or More Correct Choices	3	120	2.5
Multiple-Choice Correct Choice: NONE	2	55	3.6
Total Multiple-Choice Exercises	49	1053	4.7
Yes-No Exercises (except opinion questions)	1	133	0.8
True-False	0	254	0.0
Other Implied-Choice	0	59	0.0
Total Implied-Choice	1	446	0.2
Predicted AID Response	82	1861	4.4
Constructed AID Command	34	798	4.3
Reported Result of AID Use	14	878	1.6
Other Constructed-Response Exercises	62	1476	4.2
Total Constructed Response	192	5013	3.8
Totals	242	6512	3.7

Table 8

Comparison of Scores in Lessons with Scores in Related Exercises in Subtests

Lesson number	Proportion correct in lesson (%)	Proportion correct in test (%)
1	67.9	85.0
2	63.3	75.0
3	58.2	71.4
4	69.4	46.7
5	67.6	52.7
8	66.7	56.0
9	68.4	56.2
10	80.8	54.4
11	73.3	80.6
12	77.8	52.4
15	66.6	61.7
16	61.0	59.1
17	11.9	51.5
18	49.4	52.7
19	67.4	54.6
Mean	63.3	60.7
S.D.	16.1	11.7
N	15	15

Correlation coefficient (R) = .165

R^2 = .027

on each subtest was compared with the proportion correct on the lesson tested by that subtest. Percentages correct on these subtests are given in the second column in Table 8 and are associated with the appropriate lesson. There is essentially no correlation between percent correct on lessons and percent correct on the associated internal tests ($R^2 = .027$). The predictive power of lesson scores for individual students will be explored in a later analysis of individual performance.

Multiple-choice Exercises

In the first 21 lessons of the AID course there are 80 multiple-choice exercises. In all of these exercises the choices are listed and labeled with letters, and if there is more than one correct choice, the students are expected to type the labels for all of the correct choices. Two formats are used:

(Vertical)

A. TYPE
B. DELETE
C. SET
D. DO
N. NONE OF THE ABOVE

(Horizontal)

	A		B
1.2	SET X=1	1.2	DEMAND X
1.3	TYPE X↑2	1.3	TYPE X↑2
1.4	TYPE X↑3	1.4	TYPE X↑3

Generally, the vertical form is used if each choice can be printed on one line, and the vertical form is used if several lines are required to print single choices.

The data for multiple-choice exercises are divided into four classes, according to the number of correct choices (one, two, more than two, none). The number of correct first responses, total number of first responses, and percent correct on first responses for each problem type are shown in Table 4 presented earlier. The class of multiple-choice exercises with one correct choice is sufficiently numerous (56 exercises) to warrant a more detailed inspection of the data. These exercises were subdivided into the 10 following classes, and proportion correct was calculated for each of these classes.

(a) Algebraic Equivalence I. The student is given an algebraic expression and is asked to choose an equivalent expression.

Example:

TYPE 10/7 - 5 - 2

COULD BE WRITTEN

A. TYPE 10/(7-5) - 2
B. TYPE (10/7) - (5-2)
C. TYPE (10/7) - 5 - 2
N. NONE

(b) Algebraic Equivalence II. The student must choose a described algebraic expression.

Example:

WHICH COMMAND WILL CAUSE AID TO MULTIPLY 25 BY 5 AND DIVIDE BY 3?

A. TYPE 25 × 5/3
B. TYPE 25 * 5/3
C. TYPE 25(5/3)
N. NONE OF THE ABOVE

(c) Choice of AID Programs. The student must choose which of two routines will produce a specified result.

Example:

WHICH PART TYPES THE SMALLEST OF TWO NUMBERS?

A. 3.1 TYPE R IF R > S
   3.2 TYPE S IF R <= S

B. 4.1 TYPE R IF R < S
   4.2 TYPE S IF R >= S

(d) Mechanics. The student is asked about the mechanics of using the instructional program or the AID interpreter.

Example:

WHICH COMMAND WILL STOP THE AID INTERPRETER, AND RETURN YOU TO YOUR LESSON?

A. CTRL-H
B. RETURN
C. INST
D. CTRL-T

(e) Syntax of AID Commands. The student must decide which of a list of commands is syntactically correct.

Example:

WHICH OF THESE COMMANDS WILL CAUSE AID TO STOP AND WAIT FOR YOU TO TYPE A VALUE FOR S?

A. 3.7 ASK S = *
B. 3.7 DEMAND S
C. 3.7 REQUEST S =
D. 3.7 DEMAND S

(f) Semantics of AID Commands. The student must choose from among syntactically correct commands those that effect a specified action.

Example:

WHICH COMMAND(S) WILL NEVER GIVE A NEGATIVE NUMBER NO MATTER WHAT THE VALUE OF X IS?

A. TYPE X/5
B. TYPE SGN(X)*(X/5)
C. TYPE X/(-5)
D. TYPE SGN(X)*(X/(-5))
N. NONE

(g) Boolean Equivalence I. A Boolean statement is given, and the student must choose an equivalent statement.

Example:

X <= M

MEANS THE SAME AS WHICH OF THESE?

A. M # X
B. M <= X
C. M >= X
N. NONE

(h) Boolean Equivalence II. The student must choose a Boolean statement, given a description of it in English.

Example:

WHICH MEANS "Q IS NON-NEGATIVE?"

A. Q > 0
B. Q >= 0
C. Q < 0
D. Q <= 0
N. NONE

(i) Sequence of Execution. The choices are lists of step numbers, and the student must choose the list that executes the commands in a specified order.

Example:

```
16.1  DO STEP 2.1 IF Q < 0
16.2  TYPE Q
 2.1  SET Q = - Q
DO PART 16
```

IF Q = 3, THE STEPS WILL BE DONE IN WHICH ORDER?

A.	B.	C.	D.
16.1	16.1	16.1	16.1
2.1	16.2	2.1	16.2
16.2	2.1		

(j) Miscellaneous

The number of exercises, number of first responses, and percent correct on first responses for each of the above classes of multiple-choice exercises are given in Table 9.

It is difficult to draw firm conclusions from this subdivision of problems but these data present some clues to student behavior. The two classes with the highest proportion correct are (d) Mechanics, and (g) Boolean Equivalence I. The two classes with the lowest proportion correct are (c) Choice of AID Programs and (h) Boolean Equivalence II. The proportion correct for class (i) Sequence of Execution is only 51.6% over 13 exercises. It may be that the students have difficulty understanding order of execution of commands and that the curriculum should emphasize this area. The data from later lessons on loops will be studied to see if they also suggest some revision in this aspect of the course.

Table 9

Number of Exercises, Number and Percent of Correct First Responses in the 10 Classes of Single Choice Multiple-choice Exercises in Lessons 1 to 21 (Excluding Lessons 7, 14, 21)

Exercise type	Number of Exercises	Total correct first responses	Total first responses	Percent correct
(a) Algebraic Equivalence I	6	38	62	61.3
(b) Algebraic Equivalence II	2	12	25	48.0
(c) Choice of AID Programs	3	13	38	34.2
(d) Mechanics	6	74	88	84.1
(e) Syntax of AID Commands	7	61	91	67.0
(f) Semantics of AID Commands	2	16	31	51.6
(g) Boolean Equivalence I	3	38	49	77.6
(h) Boolean Equivalence II	2	13	34	38.2
(i) Sequence of Execution	13	97	188	51.6
(j) Miscellaneous	12	74	112	66.1
Total	56	436	718	60.7

In multiple-choice exercises, several variables other than problem type may contribute to difficulty. One of these variables is the number of choices given; presumably the larger the set of possible answers, the more difficult the choice. The range of choices in these exercises is from two to five. To estimate the effect of the number of choices, the correlation between proportion correct and number of choices was calculated, using only those exercises (49 of the 56) for which there were 10 or more first responses. This correlation was .000, indicating that the number of choices bears no linear relationship to problem difficulty. This result is not conclusive since number of choices and whether an exercise is in Class (c) or (d) are statistically dependent. Further, Class (c) is the most difficult class but has an average of only two choices per exercises, whereas Class (d) is the least difficult class but has an average of 4.3 choices per exercises.

A further analysis was made by using multiple step-wise linear regression with observed proportion correct, x_1, as the dependent variable, and using the following independent variables:

x_2 number of choices

x_3 1 if one of the choices is
N. NONE OF THE ABOVE
0 otherwise

x_4 1 if the exercise is in Class (a) Algebraic Equivalence I
0 if not

x_5 1 if the exercise is in Class (b) Algebraic Equivalence II
0 if not

x_6 1 if the exercise is in Class (c) Choice of AID Programs
0 if not

x_7 1 if the exercise is in Class (d) Mechanics

0 if not

x_8 1 if the exercise is in Class (e) Syntax of AID Commands

0 if not

x_9 1 if the exercise is in Class (f) Semantics of AID Commands

0 if not

x_{10} 1 if the exercise is in Class (g) Boolean Equivalence I

0 if not

x_{11} 1 if the exercise is in Class (h) Boolean Equivalence II

0 if not

x_{12} 1 if the exercise is in Class (i) Sequence of Execution

0 if not

x_{13} 1 if the exercise is in Class (j) Miscellaneous

0 if not.

This regression was computed for the 49 exercises for which there were 10 or more first responses. In Table 10, the variables are listed in the order in which they were "stepped" into the regression equation, and the values of R and R^2 are given. The variables in the regression accounted for 38% of the variance of the observed proportion correct. This leaves 62% of the variance unaccounted for, indicating that there are other variables, as yet unidentified, that affect problem difficulty in this group of exercises.

The two variables that entered into the regression first, x_7 and x_6, are the variables for membership in Classes (d) and (c), the same two classes that are statistically dependent on number of choices, and that the third variables to enter into the regression is x_2, the number of

Table 10

Order in Which Independent Variables Were Entered into a Multiple Stepwise Regression with the Associated Correlation Coefficients (R)

Order	Variable	R	R^2	Increase in R^2
1	x_7	0.33	0.11	0.11
2	x_6	0.40	0.16	0.05
3	x_2	0.52	0.27	0.11
4	x_5	0.54	0.29	0.02
5	x_8	0.55	0.31	0.02
6	x_3	0.56	0.32	0.02
7	x_{10}	0.59	0.35	0.03
8	x_{12}	0.60	0.36	0.01
9	x_{13}	0.61	0.37	0.01
10	x_{11}	0.61	0.38	0.01
11	x_4	0.61	0.38	0.00
12	x_9	0.61	0.38	0.00

choices. x_7 and x_6 together accounted for 16% of the variance in the observed proportion correct, whereas the addition of x_2 increased that figure by 11%, indicating that the number of choices is of significant effect even after Classes (c) and (d) are taken into account. The fact that x_2 entered into the regression before any of the other variables (except x_7 and x_6) supports the hypothesis that the number of choices contributes to problem difficulty.

Constructed-response Exercises

The constructed-response exercises form by far the most numerous class of exercises in the course (390 out of 675), and represent more diversity than either multiple-choice or implied-choice exercises. One reasonably homogeneous subclass of the constructed-response exercises, the "predicted AID response" exercises, was chosen for more detailed study. This class contains 135 exercises, and the data contains 1,861 responses to be analyzed.

In the "predicted AID response" exercises, the student is asked to predict the result if a given command were to be executed by the AID interpreter. For these exercises, a simple classification scheme is not possible because of the interaction of variables that contribute to the relative difficulty of each exercise. For this reason, multiple stepwise linear regression was used to find which of these variables had the greatest influence on problem difficulty. Only the 104 exercises with 10 or more responses were used in this study. The dependent variable, x_1, to be predicted by the regression equation was observed proportion correct. Eleven independent variables were identified and values assigned to each exercise. The independent variables were the following:

x_2 Number of arguments for TYPE commands. The problem may contain several TYPE commands, each of which can have one or more arguments. The number of arguments for all TYPE commands are summed to give the value of x_2.

x_3 Number of AID commands. All of the commands displayed in the problem are counted for x_3. If the exercise is a continuation of a preceding exercise or exercises, the commands displayed in the preceding exercise may also be counted into this variable. Variables x_3 to x_{11} are concerned with this extended set of commands.

x_4 Proportion of AID commands to be used. Some commands may be extraneous. x_4 is obtained by dividing the number of commands needed by the total number of commands displayed.

x_5 Number of function calls. x_5 is the total number of function calls in the displayed commands, including both defined functions and standard AID functions.

x_6 Number of clauses. The number of IF, FOR, and TIMES clauses are counted.

x_7 Number of substitutions required. x_7 is a count of the substitutions required for the correct solutions, including substitution of numbers for real variables and substitution of expressions for real variables. Substitution of function definitions is not counted. As an example, suppose these commands were given:

```
LET F(X) = X+2
SET A = 5
TYPE F(27/A)
```

The substitution of 5 for A is counted, and the substitution of 27/A for X in the expression X+2 is counted, so the value of x_7 is 2.

x_8 Imbeddedness of arithmetic expression. The student may be asked to evaluate some arithmetic expression, possibly after performing one or more substitutions. To measure imbeddedness, we used a "completed" expression, which is obtained from the expression displayed by making all required substitutions and by inserting all implicit parentheses. This completed expression is the basis for calculating variables x_8 and x_9. For x_8, imbeddedness is measured by finding the maximum number of left parentheses that can be counted before a right parenthesis is encountered. As an example, consider these commands:

```
SET X = 2
TYPE X + X↑3
```

The completed expression derived from this would be

```
2 + (2↑3)
```

and the imbeddedness variable x_8 would have the value 1 since there is only one pair of parentheses.

x_9 Number of operations. The value of x_9 is found by counting the number of operation symbols in the completed arithmetic expression.

x_{10} Number of exponentiation operators. The occurrences of the symbol ↑ in the extended set of displayed commands are counted for x_{10}.

x_{11} Number of implied parentheses. Here the completed arithmetic expression is not used for a basis for calculation. What is counted is the number of pairs of parentheses that are used implicitly. For example, in this command,

SET X = 2↑3/4,

there is one pair of implied parentheses, (2↑3)/4, and the value of x_{11} is one.

x_{12} Number of preceding "predicted AID response" exercises in the same lesson.

Table 11 shows the order in which the variables were "stepped" into the regression equation and the values for R and R^2. The variables used accounted for only 21% of the variance in proportion correct. The variable with the greatest effect was x_9, the number of operations. This variable is related to, but not identical with, the operations variable used by Loftus (1970) and by Jerman (1971) in similar models of problem-solving difficulty. In both cases, the operations variable was found to be significantly related to difficulty. The second most effective variable was x_6, the number of clauses, and the third was x_5, the number of function calls.

The variables x_{11}, x_2, x_7, x_3 and x_8 all added essentially nothing to the value of R^2. In fact, the first six variables accounted for 20% of the variance and the next four together added only 1% more.

8. Analysis of Selected Aid Data

Although this report is concerned primarily with data collected by INST, a large body of data was also collected by the AID interpreter, and a sample of this data was selected and analyzed by hand. These data

Table 11

Order in Which Independent Variables Were Entered into a Multiple Stepwise Regression with the Associated Correlation Coefficients (R)

Variable	R	R^2	Increase in R^2
x_9	0.30	0.09	0.09
x_6	0.41	0.17	0.08
x_5	0.43	0.18	0.01
x_{10}	0.43	0.19	0.01
x_{12}	0.44	0.19	0.00
x_4	0.44	0.20	0.01
x_8	0.45	0.20	0.00
x_3	0.45	0.20	0.00
x_7	0.45	0.21	0.00
x_2	0.46	0.21	0.00
x_{11}	0.46	0.21	0.00

were collected by the AID interpreter as students used it to solve problems given by the instructional program. The data consists simply of the characters typed by the student together with necessary bookkeeping information such as student number, problem identifiers, and date.

The problem of analyzing these data from the AID interpreter is basically unsolvable. As was discussed earlier, no effective procedure exists that can determine if a student-written program is equivalent to a given, correct program. An approximation of this procedure may be possible, but for the moment data collected by the AID interpreter must be analyzed by hand. However, no evaluation of the course will be complete without an examination of the programs written by the students. In a later report, more sophisticated and extensive analyses of the AID interpreter data will be made.

In the first 21 lessons of the course, there are 58 exercises that require the student to use the AID interpreter. These exercises vary greatly in difficulty. Some simply ask the student to call the AID interpreter and copy a single command; others ask the student to apply recently learned principles to solve complex problems. Of the 58 exercises requiring the use of AID, 28 are essentially copying tasks. Of the remaining 30 exercises, there are approximately 20 that can serve as genuine tests of the students' ability to use the AID interpreter. Eight of these exercises were chosen for analysis, the results of which are summarized in Table 12.

All these exercises follow the same form. The instructional program presents a problem and asks the student to use the AID interpreter to solve it (the student may request hints at this point, but he cannot

Table 12

Summary of Student Performance on Selected "Use AID" Exercises

Lesson no.	Problem no.	No. students who used AID	No. correct solutions: first use of AID	% correct: first use of AID	No. correct solutions: all AID uses	% correct all AID uses
5	30	13	2	15.4	12	92.3
5	31	12	3	25.0	11	91.7
8	9	20	11	55.0	15	75.0
8	27	16	4	25.0	6	37.5
8	28	12	4	33.3	12	100.0
9	3	18	3	16.7	7	38.9
12	4	15	6	40.0	9	60.0
15	15	11	2	18.2	8	72.7
Totals	8	117	35	29.9	80	68.4

give an answer until he has called the AID interpreter). Next, the student calls the AID interpreter and attempts to solve the problem. This step is called the "first use of AID" in Table 12. The student may at this time write a program, try it out, replace one or more commands, correct typographical errors, etc. Finally, when he is satisfied with his solution, the student switches back to the instructional program and is asked for a report on the results he obtained. If his reported results are incorrect, the instructional program gives him additional instruction and asks him to switch back to AID and try again.

Table 12 shows the number of students who correctly solved each problem on the first try and the number of students who eventually arrived at a correct solution. The percent correct for first AID use is quite low (29.9%). Since these problems are among the most difficult in the entire course, this percent is expected to be lower than the 66% average for other exercises. Because of the additional help given by the instructional program after the first use of AID, the percent correct is expected to rise when subsequent uses are considered, and, in fact, it does rise to 68.4%. The difference between proportion correct for first try and for subsequent tries is most pronounced for Lesson 5, Exercise 30. Only two out of 13 students solved the problem on the first try, but 12 of them solved it eventually.

During the hand-grading of these exercises, notes were made on the most common errors for each problem. Following are the problem statements (including subproblems) for each of the eight exercises, including comments on the errors observed while the exercises were being graded.

Problem 5-30:

1 CENTIMETER = .3937 INCHES. START AID AND CONVERT THE FOLLOWING LENGTHS TO CENTIMETERS:

6.9 INCHES
7.445 INCHES
23.9753 INCHES

Problem 5-30.1:

FROM THE ABOVE CALCULATIONS,

6.9 INCHES = ??? CENTIMETERS

Problem 5-30.2:

7.445 INCHES = ??? CENTIMETERS

Problem 5-30.3:

23.9753 INCHES = ??? CENTIMETERS

The students must solve this problem with direct TYPE and SET commands since indirect (stored) commands have not yet been introduced. One solution would be

```
SET X = .3937
TYPE 6.9/X, 7.445/X, 23.9753/X
```

On this problem only two of 13 students were correct on the first try. Seven of the students made algebraic errors; six used multiplication instead of division, and one used .3937/6.9 instead of 6.9/.3937. Two students made algebraic errors on the second try, one repeated the error of using multiplication rather than division, and one came up with a unique use for π (6.9/3.1416).

Although a number of syntax errors were made, they were almost all typographical errors and were corrected by the student without any

intervening instruction. Only one student was unable to correct his errors in syntax. It is likely that had the students known the correct algebraic formulation, all but one would have produced a correct solution on the first try.

Problem 5-31:

> TO FIND THE NEW AMOUNT IN A SAVINGS ACCOUNT, CALCULATE THE INTEREST AND ADD IT TO THE LAST BALANCE. START AID AND CALCULATE THE INTEREST AND THE NEW BALANCE AFTER ONE YEAR FOR AN ACCOUNT WITH AN INTEREST RATE OF 4.5 PERCENT PER YEAR AND A PREVIOUS BALANCE OF $3274.86. (ASK FOR A HINT IF YOU NEED ONE.)

Problem 5-31.1:

> WHAT IS THE INTEREST ON THE ABOVE ACCOUNT TO THE NEAREST PENNY?

Problem 5-31.2:

> WHAT IS THE NEW BALANCE IN THE ACCOUNT?

This problem can be done in several ways. One solution, which was not used by any of the students, would be

```
SET P = 3274.86
SET I = .045 * P
TYPE I, P+I
```

Only three out of 12 students were correct on their first try, but 11 eventually succeeded. Two students made syntax errors of the same kind; they both used the symbols $ and % in an arithmetic expression. Two students calculated the new balance directly, without calculating the interest independently, which left them unable to answer the question in the first subproblem. Obviously, the problem should have been explicit

in its request for two independent calculations. Four students used 4.5 as the interest rate rather than .045, and two students used the wrong arithmetic operation. Of the 11 students who eventually gave correct answers, only four used variables. Here again it would seem that the major difficulty is with algebra, rather than with the use of AID.

Problem 8-9:

> START AID AND USE A "LET" COMMAND TO DEFINE A FUNCTION THAT GIVES THE RECIPROCAL OF X. USE YOUR FUNCTION TO FIND THE RECIPROCAL OF
>
> 119.4
> 67.3↑3
> 6+4

Problem 8-9.1:

> WHAT IS THE RECIPROCAL OF 119.4?

A correct solution of this exercise would be

```
LET F(X) = 1/X
TYPE F(119.4), F(67.3↑3), F(6+4)
```

Over half of the students solved this on the first try, and three-quarters of them were eventually correct. The LET command was introduced in Lesson 8, and this was the second exercise in which the students used AID in this lesson. Five out of 20 students made syntax lerrors that they could not correct, all in the LET command. Three students made algebraic errors, using as the formula for reciprocal .1 x, $x^2/2$, and $1/x^2$. Two students who defined the function correctly redefined it needlessly before each function call.

In this exercise, the syntax of the newly introduced LET command was the source of most errors.

Problem 8-27:

> USE AID TO DO THIS PROBLEM. DEFINE A "VOLUME" FUNCTION THAT WILL GIVE THE VOLUME OF A CYLINDRICAL TANK OF RADIUS R AND HEIGHT H.
>
> (VOLUME = 3.1416 TIMES THE RADIUS SQUARED TIMES THE HEIGHT.)
>
> FIND THE VOLUMES OF 2 TANKS:
>
> > TANK A IS 57.5 FEET HIGH AND HAS A RADIUS OF 18.6 FEET.
> >
> > TANK B IS 65.4 FEET HIGH AND THE RADIUS IS 19.3 FEET.

Problem 8-27.1:

> WHAT IS THE VOLUME OF TANK B?

Problem 8-27.2:

> WHICH TANK HOLDS MORE, TANK A OR TANK B?

A correct solution for this exercise is

```
LET V(R,H) = 3.1416 * R↑2 * H
TYPE V(18.6, 57.5), V(19.3, 65.4)
```

Only four of 16 students succeeded on this exercise on the first try, and only six eventually succeeded. Seven students did not define a function at all but solved the problem arithmetically. Most of the students who tried to use a function were confused about the syntax of the LET command, the use of functions of more than one variable, or the use of dummy variables. Two students used the wrong algebraic formula (πrh instead of $\pi r^2 h$) even though the problem gave the formula.

Here it seems the course moved too fast in introducing functions of more than one variable so soon after simple functions were introduced. Some of the preceding exercises which required substitution of expressions containing real variables for the dummy variables used in function definitions may have exacerbated an existing confusion about variables. The next exercise which required a function of only one variable proved much less difficult.

Problem 8-28:

> USE AID TO DO THIS PROBLEM. DEFINE A FUNCTION TO CONVERT DEGREES FAHRENHEIT TO DEGREES CENTIGRADE. THEN CONVERT THESE TEMPERATURES TO CENTIGRADE:
>
> 0, 10, 32, 72, 212

Problem 8-28.1:

> 72 DEGREES FAHRENHEIT IS EQUAL TO HOW MANY DEGREES CENTIGRADE?

A correct solution is

```
LET C(F) = (F-32) * 5/9
TYPE C(0), C(10), C(32), C(72), C(212)
```

and one-third of the students gave this solution, or an equivalent one, on their first try. All of the students arrived at a correct solution on a later try.

Seven out of 12 students made algebraic errors, five of them identical: x - 32 * 5/9 instead of (x-32) * 5/9. Omission of necessary parentheses in an algebraic expression with only two operations indicates a basic confusion about the hierarchy of operations.

Problem 9-3:

WRITE A FUNCTION, H(A,B), THAT WILL FIND THE HYPOTENUSE OF A RIGHT TRIANGLE IF THE LENGTHS OF THE OTHER TWO SIDES ARE GIVEN BY A AND B. START THE AID INTERPRETER AND TEST YOUR FUNCTION ON THESE TRIANGLES;

1. A=3, B=4
2. A=12, B=12
3. A=1/2, B=3/4
4. A=9, B=13.2

Problem 9-3.1:

WHAT IS THE HYPOTENUSE OF THE FIRST TRIANGLE ABOVE?

(WHERE A=3, B=4)

Problem 9-3.2:

WHAT IS THE HYPOTENUSE OF THE TRIANGLE WITH SIDES A=1/2 AND B=3/4?

This is the first exercise using AID after the standard AID function SQRT is introduced, and the correct solution is

```
LET H(A,B) = SQRT(A↑2 + B↑2)
TYPE H(3,4), H(12,12), H(1/2,3/4), H(9,13.2)
```

The students did very poorly on this problem. Three of the 18 were correct on their first try, but only seven eventually achieved a correct solution. Most of the errors were syntax errors either in the LET command or in the function call. Like Problem 8-27, there were a number of errors in the use of parentheses:

1/2↑2 + 3/4↑2 was used for (1/2)↑2 + (3/4)↑2

SQRT(A↑2) + (B↑2) was used for SQRT(A↑2 + B↑2)

SQRT A was used for SQRT(A)

SQRT(A)↑2 + (B)↑2 was used for SQRT(A↑2 + B↑2)

916.25↑1/2 was used for 916.25↑(1/2)

Several students did not know the correct formula and used:

$(\sqrt{A} + \sqrt{B})^2$

$(A^2) + (B^2)$

Three students did not define a function but did produce correct specific solutions for all four triangles.

Problem 12-4:

START AID AND WRITE A PROGRAM (PART) THAT WILL ASK YOU FOR 3 NUMBERS A, B, AND C, AND THEN GIVE YOU THE AVERAGE OF THE 3 NUMBERS. AFTER YOU HAVE TESTED YOUR PROGRAM, USE IT TO FIND THE AVERAGE OF

A = 179.053
B = 23.7
C = 271.0015

Problem 12-4.1:

WHAT ANSWER DID YOU GET?

This is the first exercise in the eight hand-graded exercises that requires the use of a stored program. One correct solution is

```
4.1  DEMAND A
4.2  DEMAND B
4.3  DEMAND C
4.4  TYPE (A+B+C)/3
```

This program would be executed by giving the command

DO PART 4

Six of 15 students solved this problem on the first try, and another three succeeded later. The majority of errors were algebraic; seven students used incorrect formulas:

A + B + B/3 (used by four students)
ABC/3 (used by two students)
3/A + B + C (one student)

Of the nine correct solutions, four were not debugged, and the others were tested for only one set of poorly chosen values. Here again, as in Problem 9-3, some students produced specific but not general solutions.

Problem 15-15:

USING AID, WRITE A PROGRAM THAT WILL FIND THE SMALLER OF TWO NUMBERS X AND Y. TRY SEVERAL DIFFERENT VALUES OF X AND Y.

Problem 15-15.1:

DID YOUR PROGRAM WORK?

An economical solution to this problem is

```
9.1  DEMAND X
9.2  DEMAND Y
9.3  TYPE X IF X < Y
9.4  TYPE Y IF Y <= X
```

For this exercise, two out of 11 students were correct on their first try, and only three did not solve the problem at all.

The greatest difficulty was in inputting values for the two variables. Eight students had trouble with this. Several of them tried rather ingenious (but incorrect) variants of the FOR clause:

```
DO PART 9 FOR X = 6, Y = -2.
DO PART 9 FOR (X,Y) = (4,7), (8,4), (4321,6493).
```

Four of the eight correct solutions were well debugged, three were tested for only one set of values, and one was not tested at all.

The above discussion of the performance of students on eight selected exercises is not intended as a definitive survey or as a final evaluation of the curriculum but as an indication of the ways students solve problems and of guidelines for future analyses of student interaction with the AID interpreter. It is clear that some analysis of errors is desirable and leads to more meaningful interpretation than correct-incorrect grading alone.

In summary, the most common difficulties of the students involved algebra more than programming and centered around the following aspects:

(1) Use of parentheses. Whether or not students would have correctly formulated the algebraic expressions if they were using ordinary algebraic notation rather than AID notation is not known, but one suspects that in many cases they could not have done so.

(2) Use of standard algebraic formulas. The number of errors made in standard formulas is a strong indication that the course was delinquent in not providing these formulas wherever needed.

(3) Use of functions of more than one variable.

9. Conclusions

In this preliminary report, we discussed the interaction between students and a tutorial computer-assisted course in programming. Rates of progress as measured by the Daily Report program were discussed using one class of students as an example. A classification scheme for the exercises was devised, and the first responses to exercises in the first 21 lessons were examined by problem type and by lesson. Two classes of exercises, one set of 49 multiple-choice exercises and one set of 104

constructed-response exercises, were analyzed in detail to determine the structural variables that affected problem difficulty. Multiple stepwise linear regression was used to model exercise characteristics as determiners of exercise difficulty. Eight exercises that required the students to use the AID interpreter were analyzed by hand.

Some of the results of these investigations were unexpected, some confirmed predictions, and some were anomalous. First, the proportion of correct initial responses (65.7%) was lower than hoped for. There are no firm guidelines giving the optimal proportion correct for CAI. Some curriculum designers aim for nearly 100% correct, others have claimed 85% to be ideal. We assumed that there is little learning if the success rate is very high. On the other hand, with a very low success rate, students are likely to become discouraged. Until further research suggests an optimum success rate, each curriculum designer is left to his own best judgment. In our case, we believed that the proportion of correct first responses should be about 75%, which is nearly 10% higher than the proportion correct attained by the students studied in this investigation. It should be noted, however, that proportion correct summed over students is a simplistic measure of effective instruction, particularly for computer-assisted instruction where complex branching strategies are used, and better measures are obtained by examining the distributions of correct answers achieved by individual students over all exercises and the success with which students meet the goals of the course.

Second, proportion correct among all the lessons was quite stable, with only four lessons in the first 21 deviating from the total 65.7%

correct by more than 10%. The very low 11.9% for Lesson 17 was unexpected and remains unexplained.

Third, examination of proportion correct for different problem types indicated that the multiple-choice exercises were more difficult than the constructed-response exercises (54.9% correct compared to 66.6%).

Fourth, the multiple-choice exercises that were examined closely were those with only one correct choice, excluding the exercises for which the correct answer is "none of the above." The total proportion correct for these exercises was 60.7%, but the variation between different subclasses is striking. The easiest multiple-choice exercises concerned the mechanics of operating the instructional program and the AID interpreter; the most difficult were those in which the student had to choose which of two given AID programs would produce a specified result.

Fifth, we found no simple, unambiguous relationship between difficulty and the number of correct answers in multiple-choice exercises. The category labeled "multiple-choice, 1 correct choice" was easiest in terms of proportion of correct first responses (60.7%), but the other single choice category, "multiple-choice, correct choice: none," was the most difficult (41.8%). Further, the category with three correct choices was easier (49.2%) than the category with two correct choices (37.5%).

Sixth, a more sophisticated approach was taken in the analysis of performance on a selected class of constructed-response exercises. The exercises used were those in which the student predicts the result of using given AID commands. A dozen structural variables were defined and a stepwise linear regression performed to determine the parameters that could estimate the difficulty of these and similar exercises. We

succeeded in obtaining a model that accounted for 21% of the obtained variance in problem difficulty, a reasonable but not spectacular fit for a first model. Three structural variables (the number of operations, the number of clauses, and the number of function calls) accounted for 18% of the variance. All these variables correlated negatively with proportion correct.

The investigation suggested a few general conclusions about student behavior. The mechanics of using the instructional system did not cause undue difficulty, as shown by the performance of the students on Lesson 1 and the subsequent test of that lesson, as well as by their performance on the Multiple-choice Exercise Class. It remains unclear whether or not the students learned to use the various control features as effectively as they could (the number of requests for hints was much lower than expected), and further investigation is needed of this aspect of the instructional system.

The use of direct commands, and the concept of stored programs and their execution were readily mastered. The syntax of AID commands, except for the LET command, caused little difficulty. Although many students made syntactical errors in using the AID interpreter, most of these errors were immediately corrected by the students.

The three areas in which the students seemed least adept were:

(1) Hierarchy of arithmetic operations.

(2) Use of functions, especially functions of more than one variable.

(3) Sequence of execution of AID commands.

This last observation is substantiated by the study of structural variables in constructed-response exercises, and also by the examination of the student performance using the AID interpreter. Because of the difficulty students had with the hierarchy of operations and the use of functions, it is reasonable to assume that they lacked necessary algebraic experience and that the course did not provide that experience for them. It is harder to draw conclusions from the difficulty students had with the sequence of execution. If this result holds true for later lessons and for other groups of students, it will pinpoint a major weakness in the course, since an understanding of the sequence of execution is essential in programming. We hope that in later reports more conclusive evidence can be supplied on these and other unanswered questions.

REFERENCES

Davis, M. Computability and unsolvability. New York: McGraw-Hill, 1958.

Friend, J., & Atkinson, R. C. Computer-assisted instruction in programming: AID. Technical Report No. 164, Institute for Mathematical Studies in the Social Sciences, Stanford University, January 25, 1971.

Friend, J. INSTRUCT coders' manual. Technical Report No. 172, Institute for Mathematical Studies in the Social Sciences, Stanford University, May 1, 1971.

Jerman, M. Instruction in problem solving and an analysis of structural variables that contribute to problem-solving difficulty. Technical Report No. 180, Institute for Mathematical Studies in the Social Sciences, Stanford University, November 12, 1971.

Loftus, E. J. F. An analysis of the structural variables that determine problem-solving difficulty on a computer-based teletype. Technical Report No. 162, Institute for Mathematical Studies in the Social Sciences, Stanford University, December 18, 1970.

Mark, S. L., & Armerding, G. W. The JOSS primer. The RAND Corporation, Santa Monica, California, August, 1967.

Shaw, J. C. JOSS: Experience with an experimental computing service for users at remote typewriter consoles. The RAND Corporation, Santa Monica, California, May, 1965.

OFFICE OF NAVAL RESEARCH

PERSONNEL AND TRAINING RESEARCH PROGRAMS (CODE 458)

DISTRIBUTION LIST

6 Chief of Naval Research
Code 458
Department of the Navy
Washington, D. C. 20360

1 Director
ONR Branch Office
1030 East Green Street
Pasadena, California 91101

6 Director
Naval Research Laboratory
Washington, D. C. 20390
Attn: Technical Information Division

6 Director
Naval Research Laboratory
Washington, D. C. 20390
Attn: Library, Code 2029 (ONRL)

12 Defense Documentation Center
Cameron Station, Building 5
5010 Duke Street
Alexandria, Virginia 22314

(Continued from inside front cover)

96 R. C. Atkinson, J. W. Brelsford, and R. M. Shiffrin. Multi-process models for memory with applications to a continuous presentation task. April 13, 1966. (J. math. Psychol., 1967, 4, 277-300).

97 P. Suppes and E. Crothers. Some remarks on stimulus-response theories of language learning. June 12, 1966.

98 R. Bjork. All-or-none subprocesses in the learning of complex sequences. (J. math. Psychol., 1968, 1, 182-195).

99 E. Gammon. The statistical determination of linguistic units. July 1, 1966.

100 P. Suppes, L. Hyman, and M. Jerman. Linear structural models for response and latency performance in arithmetic. In J. P. Hill (ed.), Minnesota Symposia on Child Psychology. Minneapolis, Minn.: 1967. Pp. 160-200).

101 J. L. Young. Effects of intervals between reinforcements and test trials in paired-associate learning. August 1, 1966.

102 H. A. Wilson. An investigation of linguistic unit size in memory processes. August 3, 1966.

103 J. T. Townsend. Choice behavior in a cued-recognition task. August 8, 1966.

104 W. H. Batchelder. A mathematical analysis of multi-level verbal learning. August 9, 1966.

105 H. A. Taylor. The observing response in a cued psychophysical task. August 10, 1966.

106 R. A. Bjork. Learning and short-term retention of paired associates in relation to specific sequences of interpresentation intervals. August 11, 1966.

107 R. C. Atkinson and R. M. Shiffrin. Some Two-process models for memory. September 30, 1966.

108 P. Suppes and C. Ihrke. Accelerated program in elementary-school mathematics--the third year. January 30, 1967.

109 P. Suppes and I. Rosenthal-Hill. Concept formation by kindergarten children in a card-sorting task. February 27, 1967.

110 R. C. Atkinson and R. M. Shiffrin. Human memory: a proposed system and its control processes. March 21, 1967.

111 Theodore S. Rodgers. Linguistic considerations in the design of the Stanford computer-based curriculum in initial reading. June 1, 1967.

112 Jack M. Knutson. Spelling drills using a computer-assisted instructional system. June 30, 1967.

113 R. C. Atkinson. Instruction in initial reading under computer control: the Stanford Project. July 14, 1967.

114 J. W. Brelsford, Jr. and R. C. Atkinson. Recall of paired-associates as a function of overt and covert rehearsal procedures. July 21, 1967.

115 J. H. Steiger. Some results concerning subjective probability structures with semiorders. August 1, 1967

116 D. E. Rumelhart. The effects of interpresentation intervals on performance in a continuous paired-associate task. August 11, 1967.

117 E. J. Fishman, L. Keller, and R. E. Atkinson. Massed vs. distributed practice in computerized spelling drills. August 18, 1967.

118 G. J. Groen. An investigation of some counting algorithms for simple addition problems. August 21, 1967.

119 H. A. Wilson and R. C. Atkinson. Computer-based instruction in initial reading: a progress report on the Stanford Project. August 25, 1967.

120 F. S. Roberts and P. Suppes. Some problems in the geometry of visual perception. August 31, 1967. (Synthese, 1967, 17, 173-201)

121 D. Jamison. Bayesian decisions under total and partial ignorance. D. Jamison and J. Kozielecki. Subjective probabilities under total uncertainty. September 4, 1967.

122 R. C. Atkinson. Computerized instruction and the learning process. September 15, 1967.

123 W. K. Estes. Outline of a theory of punishment. October 1, 1967.

124 T. S. Rodgers. Measuring vocabulary difficulty: An analysis of item variables in learning Russian-English and Japanese-English vocabulary parts. December 18, 1967.

125 W. K. Estes. Reinforcement in human learning. December 20, 1967.

126 G. L. Wolford, D. L. Wessel, W. K. Estes. Further evidence concerning scanning and sampling assumptions of visual detection models. January 31, 1968.

127 R. C. Atkinson and R. M. Shiffrin. Some speculations on storage and retrieval processes in long-term memory. February 2, 1968.

128 John Holmgren. Visual detection with imperfect recognition. March 29, 1968.

129 Lucille B. Mlodnosky. The Frostig and the Bender Gestalt as predictors of reading achievement. April 12, 1968.

130 P. Suppes. Some theoretical models for mathematics learning. April 15, 1968. (Journal of Research and Development in Education. 1967, 1, 5-22)

131 G. M. Olson. Learning and retention in a continuous recognition task. May 15, 1968.

132 Ruth Norene Hartley. An investigation of list types and cues to facilitate initial reading vocabulary acquisition. May 29, 1968.

133 P. Suppes. Stimulus-response theory of finite automata. June 19, 1968.

134 N. Moler and P. Suppes. Quantifier-free axioms for constructive plane geometry. June 20, 1968. (In J. C. H. Gerretsen and F. Oort (Eds.), Compositio Mathematica. Vol. 20. Groningen, The Netherlands: Wolters-Noordhoff, 1968. Pp. 143-152.)

135 W. K. Estes and D. P. Horst. Latency as a function of number of response alternatives in paired-associate learning. July 1, 1968.

136 M. Schlag-Rey and P. Suppes. High-order dimensions in concept identification. July 2, 1968. (Psychom. Sci., 1968, 11, 141-142)

137 R. M. Shiffrin. Search and retrieval processes in long-term memory. August 15, 1968.

138 R. D. Freund, G. R. Loftus, and R.C. Atkinson. Applications of multiprocess models for memory to continuous recognition tasks. December 18, 1968.

139 R. C. Atkinson. Information delay in human learning. December 18, 1968.

140 R. C. Atkinson, J. E. Holmgren, and J. F. Juola. Processing time as influenced by the number of elements in the visual display. March 14, 1969.

141 P. Suppes, E. F. Loftus, and M. Jerman. Problem-solving on a computer-based teletype. March 25, 1969.

142 P. Suppes and Mona Morningstar. Evaluation of three computer-assisted instruction programs. May 2, 1969.

143 P. Suppes. On the problems of using mathematics in the development of the social sciences. May 12, 1969.

144 Z. Domotor. Probabilistic relational structures and their applications. May 14, 1969.

145 R. C. Atkinson and T. D. Wickens. Human memory and the concept of reinforcement. May 20, 1969.

146 R. J. Titiev. Some model-theoretic results in measurement theory. May 22, 1969.

147 P. Suppes. Measurement: Problems of theory and application. June 12, 1969.

148 P. Suppes and C. Ihrke. Accelerated program in elementary-school mathematics--the fourth year. August 7, 1969.

149 D. Rundus and R.C. Atkinson. Rehearsal in free recall: A procedure for direct observation. August 12, 1969.

150 P. Suppes and S. Feldman. Young children's comprehension of logical connectives. October 15, 1969.

(Continued on back cover)

(Continued from inside back cover)

151 Joaquim H. Laubsch. An adaptive teaching system for optimal item allocation. November 14, 1969.
152 Roberta L. Klatzky and Richard C. Atkinson. Memory scans based on alternative test stimulus representations. November 25, 1969.
153 John E. Holmgren. Response latency as an indicant of information processing in visual search tasks. March 16, 1970.
154 Patrick Suppes. Probabilistic grammars for natural languages. May 15, 1970.
155 E. Gammon. A syntactical analysis of some first-grade readers. June 22, 1970.
156 Kenneth N. Wexler. An automaton analysis of the learning of a nimiature system of Japanese. July 24, 1970.
157 R. C. Atkinson and J.A. Paulson. An approach to the psychology of instruction. August 14, 1970.
158 R.C. Atkinson, J.D. Fletcher, H.C. Chetin, and C.M. Stauffer. Instruction in initial reading under computer control: the Stanford project. August 13, 1970.
159 Dewey J. Rundus. An analysis of rehearsal processes in free recall. August 21, 1970.
160 R.L. Klatzky, J.F. Juola, and R.C. Atkinson. Test stimulus representation and experimental context effects in memory scanning.
161 William A. Rottmayer. A formal theory of perception. November 13, 1970.
162 Elizabeth Jane Fishman Loftus. An analysis of the structural variables that determine problem-solving difficulty on a computer-based teletype. December 18, 1970.
163 Joseph A. Van Campen. Towards the automatic generation of programmed foreign-language instructional materials. January 11, 1971.
164 Jamesine Friend and R.C. Atkinson. Computer-assisted instruction in programming: AID. January 25, 1971.
165 Lawrence James Hubert. A formal model for the perceptual processing of geometric configurations. February 19, 1971.
166 J. F. Juola, I.S. Fischler, C.T. Wood, and R.C. Atkinson. Recognition time for information stored in long-term memory.
167 R.L. Klatzky and R.C. Atkinson. Specialization of the cerebral hemispheres in scanning for information in short-term memory.
168 J.D. Fletcher and R.C. Atkinson. An evaluation of the Stanford CAI program in initial reading (grades K through 3). March 12, 1971.
169 James F. Juola and R.C. Atkinson. Memory scanning for words versus categories.
170 Ira S. Fischler and James F. Juola. Effects of repeated tests on recognition time for information in long-term memory.
171 Patrick Suppes. Semantics of context-free fragments of natural languages. March 30, 1971.
172 Jamesine Friend. Instruct coders' manual. May 1, 1971.
173 R.C. Atkinson and R.M. Shiffrin. The control processes of short-term memory. April 19, 1971.
174 Patrick Suppes. Computer-assisted instruction at Stanford. May 19, 1971.
175 D. Jamison, J.D. Fletcher, P. Suppes and R.C. Atkinson. Cost and performance of computer-assisted instruction for compensatory education.
176 Joseph Offir. Some mathematical models of individual differences in learning and performance. June 28, 1971.
177 Richard C. Atkinson and James F. Juola. Factors influencing speed and accuracy of word recognition. August 12, 1971.
178 P. Suppes, A. Goldberg, G. Kanz, B. Searle and C. Stauffer. Teacher's handbook for CAI courses. September 1, 1971.
179 Adele Goldberg. A generalized instructional system for elementary mathematical logic. October 11, 1971.
180 Max Jerman. Instruction in problem solving and an analysis of structural variables that contribute to problem-solving difficulty. November 12, 1971.
181 Patrick Suppes. On the grammar and model-theoretic semantics of children's noun phrases. November 29, 1971.
182 Georg Kreisel. Five notes on the application of proof theory to computer science. December 10, 1971.
183 James Michael Moloney. An investigation of college student performance on a logic curriculum in a computer-assisted instruction setting. January 28, 1972.
184 J.E. Friend, J.D. Fletcher and R.C. Atkinson. Student performance in computer-assisted instruction in programming. May 10, 1972.