A COMPUTER ASSISTED DRILL AND PRACTICE SYSTEM FOR INTRODUCTORY STATISTICS INSTRUCTION

THESIS

Presented to the Graduate Council of the
North Texas State University in Partial
Fulfillment of the Requirements

For the Degree of

MASTER OF ARTS

By

Richard George Ellis, B. A.
Denton, Texas
May, 1979

Ellis, Richard George, *A Computer Assisted Drill and Practice System for Introductory Statistics Instruction*. Master of Arts (Experimental Psychology), May, 1979, 86 pp., 14 tables, references, 76 titles.

To supplement college introductory statistics instruction, an interactive drill system was developed and implemented on a Hewlett-Packard 2000 timesharing computer. Unlimited practice in basic procedures and algorithms was provided over 38 topics including Chi-square, correlational methods, and one-way analysis of variance. Validation of intermediate computational steps was provided, and more difficult or remedial problems sets were made available. Optional files recorded performance data. Four support programs initialized performance files and generated summary reports. Extensive documentation and a library of reusable subroutines were designed to assist future authors to expand the system. The drill and practice system was made generally available to all university departments and students.

TABLE OF CONTENTS

TABLE OF CONTENTS--Continued

LIST OF TABLES

A COMPUTER ASSISTED DRILL AND PRACTICE SYSTEM FOR INTRODUCTORY STATISTICS INSTRUCTION

The widespread interest that has developed in this country and abroad in using the computer for instructional purposes is attested to by the volume of literature (Finch, 1972b; Kurshan, 1974; Swanson, A. K., 1973, 1974) and the number of available programs dedicated to those purposes (Lekan, 1970, 1971; Wang, 1976). Kearsley (1976) noted that Wang's 1976 Index to Computer Based Learning listed 1,837 available programs covering 137 different subjects and originating from 219 sources. This particular index, although perhaps the best available, was representative but not complete.

Psychologists have used computer technology for a range of purposes including data collection and analysis (Bailey & Polson, 1975; Sidowski, 1975). Less commonly, instructional programs have been utilized to supplement departmental offerings in statistics, research design, and physiological psychology (Snyder, 1977).

The purposes of this review were to examine computer-assisted instruction in its educational context, sample instructional efforts which use the computer to supplement or replace traditional classroom instruction in statistics, and outline some of the variables believed to be important in computer-assisted instruction.

Lippert (1971) outlined several educational uses for the computer. These included providing (a) facilities for use by students in completing homework assignments, (b) remote job entry points in laboratories and offices, (c) statistical packages for data analysis, (d) counseling and guidance services, (e) programming support for nonprogramming faculty, (f) accessible student records for placement and remediation purposes, (g) computer-managed instruction, and (h) computer-assisted instruction including facilities for automatically administering and scoring tests. Within the classroom setting, computer facilities were to be compared and evaluated along with other instructional technologies including television, radio, printed material, film, microfiche, and audio/video tape systems (Gulliford & Blau, 1976). Each technology was seen to offer positive and negative features in terms of cost, availability, and effectiveness.

Computer-based instruction obviously required a rather considerable capital investment, was unavailable in settings without appropriate programs, and was questionably cost-effective. In a lengthy survey of alternative instructional technologies, Jamison, Suppes, and Wells (1974) concluded that computer-assisted instruction, when used as a replacement for more traditional classroom instruction, was about as effective as the latter. Certain configurations of such materials (drill and practice, simulations) seemed to significantly and effectively supplement classroom instruction.

The history of computer-based instruction was relatively brief. A commonly cited precursor to modern computer instruction was the 1926 "teaching machine" of psychologist Sidney Pressey at Ohio State University which presented material, asked multiple-choice questions, evaluated student responses, and reported the correctness of answers to the student (Gulliford & Blau, 1976). Skinner (1954, 1958, 1968) attempted to translate operant learning principles into an educational technology via mechanical teaching machines and programmed instruction. In the 1960s, programmed instructional methods, electronic computers, and federal money merged to produce a number of expensive, feasibility-testing systems (Atkinson & Wilson, 1968), which demonstrated that computer-based instruction was indeed a possible and promising idea. Toward the end of that decade, the relative expense of computer-assisted instruction (CAI) gave rise to a second major trend: computer-managed instruction (CMI). In the 1970s, commercially available microcomputer systems enabled most school systems to bring computing into the public school classroom.

CMI systems developed simultaneously in several locations as a reaction to and a substitute for the earlier, expensive CAI systems exemplified by the work at Stanford (Atkinson, 1968; Suppes, Jerman, & Brian, 1968; Suppes & Morningstar, 1972). As an alternative, computer-based instructional management systems (CBIMS) did not present coursework. Rather, they offered to classroom teachers various clerical and statistical services. The typical CBIMS programs (a) pretested students to determine their standing in relationship to certain behavioral objectives, (b) diagnosed educational deficiencies, (c) prescribed assignments and activities, (d) posttested after assigned work was completed, and (e) generated summary reports for the teacher (Baker, 1971; Brudner, 1968; Finch, 1972a; Lippey, 1975; Spuck, Hunter, Owen, & Belt, 1975). While most of these systems were geared toward public schools, a few have been used to advantage in higher educational settings (Bruell, 1976; Kelly & Anandam, 1976).

In contrast to instructional management systems, the CAI programs developed were directly instructional. They may not have generated performance statistics for teachers, given and scored tests, and so forth. Instead, they were designed to actually present material and to interact with students.

CAI has not become a mainstay of academia as early proponents had predicted. Anastasio (1974) reported that developmental costs, lack of clear objectives, limited author incentives, and fear of changing social roles were some of the reasons that CAI curricula had not more rapidly diffused. It is likely that some of the inhibitions cited by Anastasio will decrease in importance with the retirement of older and less satisfactory programs (Kearsley, 1976), a reduction in computer hardware costs, and the development of easy authoring systems and languages (Alpert & Bitzer, 1970; Dowsey, 1974;

Dwyer, 1972; Faust, 1974; Frederick, 1974; Frye, 1969; Hewlett-Packard Company, 1975; McLean, 1973; Paloian, 1974; PLATO IV Software Group, 1974; Zinn, 1969).

The rationale Skinner (1968) presented for his programmed instruction included the specific benefits provided by a teaching machine such as feedback for errors and consistent, immediate contingent reinforcement for correct responses. He thought that programmed instruction was essentially a scheme for making an effective use of reinforcers. With a room filled with students, the classroom teacher could usually deliver only sporadic or delayed reinforcement. Later writers who discussed CAI echoed these ideas.

Several other aspects of CAI instruction have been noted as offering an environment which allowed self-paced progress through standard course materials. Students were able to pursue subject matter whether or not a teacher knowledgeable in that area were available (Hall, 1974). Students were encouraged to actively compose answers. Responses were evaluated and results presented to the user immediately. Relieved from time-consuming homework correction and drill in mechanical skills, teachers were freed for other pedagogical pursuits. Suppes et al. (1968) pointed out that machines removed students from socially embarassing public correction and criticism. Lastly, data collected during the course of lengthy CAI projects might shed valuable light on the learning processes themselves.

In his work, Skinner (1958) emphasized that a program or lesson should be a carefully designed sequence of steps, a frame-by-frame linear development of the desired behavioral repertoire. Incremental steps were to be optimally designed by the instructor (author) so that success at each step was assured. The analog to experimental "shaping" was obvious and perhaps a bit extreme.

In contrast, other authors emphasized increased individualization of instruction through adaptive, response-sensitive and learner-controlled CAI logical structures that more nearly personalized instruction according to the needs or desires of each student. For example, Dwyer (1971, 1974) described two "modes" of computer-based learning. Characterized as "teacher-controlled," the "dual mode" encompassed the traditional CAI methods of drill and practice, frame-oriented tutoring, testing, and CMI. As students became increasingly sophisticated at manipulating the computer, they acceded into the "solo mode" in which they developed their own models and wrote their own programs. Intrigued with their projects, Dwyer's Pittsburg high school students voluntarily spent many additional hours working with their interactive terminals.

Four instructional strategies distinguished by Kearsley (1976) have been substantially used by CAI programs designed to aid the student in learning statistics. Long, complicated programs or entire courses have utilized some or all of these instructional strategies.

Tutorial programs have presented substantial amounts of didactic material in an interactive, conversational manner. Students were quizzed and scored on-line. Simple to complex branching schemes were developed which responded to student response histories. Most easy authoring systems, such as the Hewlett-Packard Instructional Dialogue Facility, used this strategy. These tutorial programs were designed to substitute for traditional classroom instruction.

Drill and practice programs presented problems or questions to students and evaluated solutions or answers with didactic material being presented by some other means (e.g., the teacher). Branching was possible to "hints," to intermediate solutions, or to more difficult or easier examples. In 1976, this one strategy predominated all others in use, possibly because of the clearer demonstrations of its effectiveness.

A problem-solving instructional strategy utilized the computer as a tool. The user supplied problem data for various analyses. Computations were computer-generated and displayed for interpretation. Any didactic material was probably incidental or supplemental. When used for instructional purposes, "canned" programs (such as the Statistical Package for the Social Sciences) were used to analyze real or homework data.

Simulation programs presented didactic material in graphic form or modeled lifelike processes, simulating theoretical principles, research designs, or other processes.

Tubb (1977) elaborated several major permutations of computer related statistics instruction. Aside from programming courses which taught statistics as a content area, and statistics courses which taught programming as a major portion of the curriculum, computer related instruction in statistics took two general forms: hands-on and hands-off. Wegman and Gere (1972) strongly advocated a hands-off approach because of student immaturity and operational expense. They suggested that computer generated graphic and numerical demonstrations, illustrations, and examples could more appropriately be generated once for the instructor and thereafter be kept in the statistics laboratory. Drill and instruction, they felt, were less expensively accomplished by workbooks or programmed instruction texts.

A more demanding hands-on approach was to introduce statistics students to Fortran or a simpler, interactive language (such as APL or BASIC) with which they were required to program solutions to homework problems. Prewritten subroutines provided the generation of data (Tanis, 1973) and analysis of data (Koh, 1970). Advanced students at the University of Massachusetts, for example, have written programs for topics such as factor analysis and multiple regression (Abranovic, Ageloff, & Frederick, 1972). Simple interactive languages, such as OMNITAB, reduced the programming task to a minimum (Swanson, Ledlow, & Harris, 1972; Swanson, Riederer, & Weekly, 1973; Tubb, 1977). Students were able to effectively use such systems in the same session in which they were introduced to them.

Prewritten data analysis programs and subroutines have provided nonprogramming students with interpretable analyses from their own data while saving them tedious computational effort. Batch mode programs (Applebaum & Guthrie, 1970; Wikoff, 1970) and on-line programs (Williams, 1977) have been used. An interesting alternative has been the conversational, user-controlled analysis programs which asked the student to specify desired calculations or displays (Diegert, 1974; Edgar, 1973). Using prewritten routines, students could analyze considerably more data in one semester than would be possible by hand. Also, the future psychologist (businessman, educator) could practice the kind of data analysis he would most likely perform as a professional.

Learning to program a computer or to use prewritten statistical packages were efforts to avoid tedious arithmetical hand computations by calling on the computer's prodigious capacity and speed. However, ability to use "canned" programs by no means assured that the student had adequate familiarity with the computational algorithms. To address this aspect of statistical instruction, several authors have used the computer to generate homework problems for students in a manner analogous to the drill and practice CAI programs previously outlined. Wang (1976) listed 3 of 33 programs as drill and practice in whole or in part. At the University of Illinois, Avner developed an operational drill and practice program entitled Statistics A which is written in Tutor (Plato IV System) and designed as a statistical laboratory. Therios, at the University of Wisconsin, wrote a Fortran language program entitled Statistics Quiz which drilled students over course contents by randomly generating quiz questions. Cochran of East Texas State University developed a program entitled Stat in Coursewriter III which tutored, solved various problems, and drilled students over a wide range of basic statistical concepts.

Homework sample data has been randomly generated and accessed from prestored files (Count, 1969). Garrett (1970) described a Fortran subroutine which accessed any of 10 distributions and provided each student with a unique sample of data. Subsequently, students analyzed the generated data by hand. Ashburn (1977) employed a prestored file of 1,000 eighth-grade student records. Students in his class chose 30 random numbers which served as input to a master Fortran program. Each student then received serial numbers corresponding to his or her "sample" of eighth-graders. The unique data were then used to solve common, assigned problems.

Anderson, Standiford, and Alessi (1977), using Fortran, developed a computer-assisted problem solving system which administered homework problem data sets designed to develop students' computational, estimation, and procedural skills related to important concepts of an introductory statistics course. Unique problem data were generated for each student and relevant "hints" and other aids were provided when errors occurred. A calculator subroutine was discontinued after only 10% of the students were found to use it regularly. Originally on-line, all problems were eventually required to be solved off-line. Quizzes over subject matter were administered by the Plato IV System.

Wang (1976) classified 16 of 31 statistics programs as tutorial CAI in whole or in part. Two additional examples (Forsythe & Bleich, 1973; Sherr, Tuggle, & Fitch, 1974) showed how the limited graphic capabilities of a printer terminal could clarify concepts such as confidence intervals. Knief and Cunningham (1976) and Wassertheil (1969) reported positive gains for their students from their tutorial CAI projects.

While previously mentioned CAI programs supplemented traditional classroom instruction, other designs completely eliminated classroom teaching. Using BASIC and the Hewlett-Packard Instructional Dialogue Facility, Howze (1973) created a complete system of interactive instruction, homework assignments, and testing. The teacher became a tutor in special-help laboratory sessions. Grubb (1969) suggested that true "learner-controlled" tutorial systems would provide a student with "maps" of available topics which could be accessed in any order according to the needs or desires of the individual student. In this manner, an instructor might discover a student-organized "modal path" through the curriculum which varied substantially from that suggested by the instructor.

A final approach to computer related instruction in statistics has been loosely termed "simulation." Wegman (1974) suggested that computer graphics could effectively aid in developing a student's intuitive ability to visualize various distributions and other mathematical concepts. Programs have been designed to generate and sample populations, and to analyze those samples in order to demonstrate statistical principles (Cooley, 1969; Moore, 1973; Rubner, Behr, & Baker, 1974). Developed primarily for courses in research design, the EXPER SIM system (Miller, 1976) and closely related systems, such as WRIST (Snyder, 1977; Spelt & Schafer, 1976), simulated experiments based on internal models of behavioral processes and student design strategies. Outcome data were generated which the student then analyzed. Many design alternatives might be tried. A similar program was discussed by Ellsworth (1976).

While the preceding descriptions of instructional programs indicated the feasibility of computer-assisted instruction in statistics, the advisability of implementing such a system must be considered in terms of what has been learned about the effectiveness of CAI and student reactions to such programs.

Reviewing 19 evaluations of CAI programs, Jamison et al. (1974) drew three major conclusions: (a) tutorial CAI programs were about as effective as traditional instruction when used as a substitute; (b) CAI approaches often resulted in substantial time savings to learn the material; and (c) as supplemental practice, CAI particularly seemed to help disadvantaged or remedial students. It was probable that many forms of sustained, supplemental help would improve student school performance. In their evaluation of the Stanford drill and practice arithmetic program in California for 1966-67, Suppes and Morningstar (1972) noted that the teachers and administrators of a control (no-CAI) school instituted 25 minutes of extra drill and practice each day for grades four and five after receiving disappointing results from a pretest of arithmetic skills. At the end of the year, this school's students out-performed pupils in a matched school who had received 5-8 minutes of drill and practice on the computer every day.

Citing 33 sources, Edwards, Norton, Taylor, Van Dusseldorp, and Weiss (1974) presented several conclusions: (a) CAI generally resulted in increased achievement when used as a supplement to traditional instruction; (b) drill and practice CAI was the most consistently effective approach; (c) CAI was comparably effective to other forms of nontraditional educational strategies (tutors, programmed instruction); (d) CAI tutored students possibly retained less than students tutored with other methods, although only three conflicting studies were reported; (e) savings in learning time generally occurred; (f) effectiveness of computer-assisted instruction was possibly related to student sex, favoring males; and (g) student and faculty attitudes toward CAI were generally favorable.

Many project reports failed to include more than subjective evaluations of the effectiveness of implemented systems. However, several of the previously cited CAI programs in statistics and research design attempted objective evaluations. An early study (Wassertheil, 1969) with a supplemental tutorial program produced nonsignificant but suggestive improvements in examination scores. Two days before an examination, Wegman (1974) showed several computer generated graphs to an experimental group for 19 minutes. The experimental group then outperformed control subjects in the same class on that examination, but the effect disappeared on another examination given 3 weeks later. In other studies, learning achievement was improved when supplemental CAI was provided (Knief & Cunningham, 1976; Sherr et al., 1974). With the exception of Suppes and Morningstar's (1972) evaluation of the Stanford project, all other evaluations were typically small group studies done by a researcher who taught or was connected to the course. Although strongly suggestive of CAI effectiveness, the results of such studies were not conclusive.

A last major consideration was the effect of computer related instruction on student attitudes, both toward the machine and toward themselves. In an excellent and extensive review, King (1975) surveyed the relevant literature and summarized the opinions of leading contributors to the field. She concluded that computer-based instruction did not lead to feelings of "depersonalization" or "dehumanization." Rather, effective computer-based instruction could produce positive attitudes in both students and instructors, and could provide an opportunity to improve student/teacher relationships. More specifically, King identified the following series of important system variables which affected student attitudes.

Immediate feedback and reinforcement. Feedback provided a method for assessing progress and correcting errors. A related study (Tait, Hartley, & Anderson, 1973) found greater achievement in feedback as opposed to no-feedback groups but failed to show an advantage for "active" or "passive" feedback. According to King, several studies on programmed instruction indicated that a condition of intermittent reinforcement might be optimal.

Computer acceptance of alternative correct responses. Program inflexibility in accepting trivially wrong but basically correct answers frustrated students. Explicitly stated alternative answers or tolerance limits for numerical responses could be built into instructional programs. Many easy authoring systems were designed to create programs with such alternatives.

Response time. First, unpredictable or long terminal response times were irritating and disturbing. Secondly, students generally did not read extensive terminal output until it was completely printed or displayed, a finding which suggested that long, didactic prose passages might lead to inattention. Lastly, if the computer continued with further instruction too soon after verifying a student response, the student might not have adequate time to consolidate knowledge of the previous material.

Downtime. Students with positive attitudes toward the computer found downtime disappointing. Students with negative attitudes found the machine failure annoying.

King also identified the following individual difference variables, with many seeming relevant to any instructional setting.

Individual performance in relation to peer performance. Students wanted to know where they stood. The individualized self-paced nature of many CAI programs sometimes impeded an accurate assessment of class standing. Mastery of the material rather than class standing should be stressed.

Learner control. Students preferred learner-controlled programs but did not necessarily achieve more with them. Better students may have profited from such logic more than less-able students.

Anxiety. First, state anxiety was related to negative attitudes. Secondly, some students needed instructors to explain material or to answer questions. Lastly, computer-based learning may have reduced "fear of failure" by removing students from the subjective evaluations of teacher and peers in a traditional classroom.

Level of performance. Students who performed well generally had better attitudes toward computer-based instruction. Skinner's suggestion that programs try to maximize success was relevant here.

Volunteerism versus nonvolunteerism. Volunteers began with more positive attitudes toward the machine but nonvolunteers also were typically positive.

Orientation and initial contact. Because students have been initially tense and uncertain, a systematic orientation to a CAI program was felt to be appropriate to overcome any "machine shyness." King also suggested the importance of not raising expectations unduly high by overselling the value of the computer approach.

Undoubtedly, other variables will be identified and examined as computer related learning continues to evolve. For example, Cunningham (1975) suggested that while a programmed instructional format in the Skinnerian sense was better for students with low statistical aptitude, a prose style format was better for more capable students. Until a proven, detailed instructional theory was available, the set of instructional variables set forth here and elsewhere were perhaps an adequate initial guideline for producing computer-based instructional programs.

Purpose

The present project is an automated, learner-controlled system of BASIC computer programs referred to as the Drill and Practice in Statistics (DPST) System. It provides student users with drill and practice over 38 statistical procedures and algorithms encountered in introductory statistics courses at the graduate or undergraduate level. As outlined in the introduction, drill and practice CAI has proved to be effective in increasing student achievement when used as a supplement to traditional classroom instruction. The effectiveness of the approach appears to derive from the combined effects of (a) immediate reinforcement and feedback, (b) self-pacing to match presentation speed more closely to student abilities, (c) individualization of content through a learner-controlled logic which allows students to skip familiar topics in preference for unfamiliar algorithms, and (d) active student involvement with the computational aspects of the subject matter.

The DPST System is designed to provide student users with services aimed at increasing their facility with various computational procedures. Specifically, the system provides (a) personalized drill over statistical procedures and algorithms through a learner-controlled logic, (b) limited didactic material pertaining to each procedure or algorithm, (c) functionally unlimited and unique computer-supplied problem data sets, (d) remedial or more difficult problems, (e) intermediate answer checks and correction, and (f) optional student performance summaries available to the instructor at any time. Use of the system should provide more class or laboratory time for theoretical developments and explanation. Additionally, students receive an introduction to hands-on computing and may take advantage of other system software to learn the BASIC language or to use the Statistical Package for the Social Sciences. The DPST System is organized so that an unlimited number of users in various departments (Psychology, Mathematics, Education) may concurrently utilize the programs.

An attempt is made throughout the DPST System to organize the program code in a consistently structured manner. Subroutines are used where feasible and variable naming conventions are established and utilized. The functional structure of the system is uniform and self-explaining.

Program validation or testing consists of three distinct phases: (a) incorrect responses are entered for each question in the system and the machine-generated correct answers are checked by calculator; (b) all potential run-time branching linkages between programs and program segments are exercised; and (c) hand-calculated correct answers are entered for each question and validated by the system software. In addition, an error-handling routine detects random program errors and recycles the student user to an appropriate table of contents.

The DPST System physically consists of an integrated set of 16 programs, three tables, two optionally available performance files, and a library of common subroutines. The source code for the entire system consists of approximately 7,300 lines of BASIC and occupies 373 records (256 words per record). Approximately 95% of the source code is executable with the remainder consisting of remarks. The largest independently executable module consists of more than 9,000 words or 36 records of code. The optional performance files are not included in these statistics. Queries concerning the availability of the program listings or documentation should be directed to the North Texas State University Computing Center.

Instructions to Users

The Drill and Practice in Statistics System (DPST) is a coordinated series of computer-assisted instruction (CAI) drills, support programs, and files designed to provide users with unlimited practice on a variety of statistical procedures and algorithms taught in elementary statistics courses. The initial set of 38 topics (see Appendix A for list) include drills on descriptive statistics, table look-up, correlational techniques, and inferential statistical tests including one-way analysis of variance. The DPST System is implemented on the interactive Hewlett-Packard 2000 Computer at North Texas State University and may be executed from a cathode ray tube (CRT) or hard-copy terminal.

A user-controlled drill sequencing scheme is employed throughout the DPST System to enable users to select, omit, or review topics in any order. Within each topic, a user may select either descriptive material concerning the statistical procedure or unique, randomly generated problem data. Intermediate computations by users are immediately validated or corrected so that subsequent computations which depend on previous results may be revised. Performance summaries for each topic are displayed for the user. Remedial (easier) or more difficult problem data sets are available in many topics where appropriate. Optional performance statistics can be maintained in files for later perusal by instructors or researchers. Self-explaining support software initializes performance files and prints reports on demand for instructors. No programming skills are required to use to DPST System.

The initial execution procedure for the CAI drills differs slightly depending on whether a student is a casual user whose performance need not be recorded for class records, or a required user whose performance is to be recorded. The user must initially log on normally to an individual account (see Appendix B for procedure) and enter:

```
EXE-DPST00.A900
```

to execute the DPST System. After the introduction to the system, the user is queried:

```
ARE YOU TAKING THIS DRILL AS A CLASS REQUIREMENT?
(YES OR NO)
```

The casual user must enter NO or N, and then his or her first name when prompted by the machine. The required user must enter YES or Y to the question and is subsequently prompted for (a) a valid Student Record Number and (b) a nine digit Social Security Number (embedded blanks, dashes, or other nondigit characters are permissable). The required user is then asked to verify his or her last name, which the system retrieves from the performance files. If the required user enters the correct numbers and verifies the last name, the system accepts him or her as an authorized required user and proceeds normally. All users should follow further prompts and directions.

If the required user fails to enter the correct Student Record Number and Social Security Number, or fails to verify the retrieved last name, the program displays a message indicating there are identification errors in the performance files that require the instructor's attention. Subsequently, the user is automatically logged off the system.

If class performance files have not been created for the student's class, the program informs the user of the problem and proceeds as though the user were a casual user. No performance data are recorded.

Drill Sequencing and Operation

The user is allowed a maximum flexibility in sequencing the materials in the DPST System. There are three types of branching points which allow access to all parts of the system.

The unit index (see Appendix A) is initially presented to the user after introduction to the system. The user responds by entering the numeric code for any listed unit or option.

The system responds to a unit selection by chaining to the requested drill program. Immediately, the unit name and a topic index associated with that unit are displayed. The topic index consists of the titles of the specific drills in the unit. Additionally, the user is allowed the option of returning to the unit index to select a different unit. In both the unit index and the specific topic indexes the user is provided the option of terminating the drill session.

In response to a topic selection, the system initiates the specific drill requested by the user and immediately displays the topic title and four additional choices: descriptive material, problem data, topic index, and log off. The descriptive material includes a brief discussion of the selected algorithm or procedure. A computational scheme or formula is usually presented along with a solved example. The user is allowed ample time to read the text, copy the formula, or rework the solved example. After the didactic material is completed, a problem data set is randomly generated for the user.

Problem data may be initially selected as the type of instruction for a topic-attempt. In this case, the descriptive material for the topic is skipped. A problem data set is generated and displayed. A set of calculations for the user to perform is listed. The system then pauses until the user indicates a readiness to continue. It is desirable at this point for the user to copy the data onto a piece of paper.

After the user signals a readiness to continue (by pressing RETURN) the DPST System asks for the value of a specific quantity (e.g., mean, standard deviation). The user responds by entering a calculated numeric constant for that quantity. If the user's answer is correct, a complimentary message to that effect is displayed. Otherwise, an error message appears along with the correct answer. A formula or other corrective hint may be displayed. Whether or not the user's answer is correct, the system usually pauses to allow the student to verify the correct answer or to compute the next required value. The sequence of question-answer-analysis-pause is repeated for all questions asked during a topic (typically five or six questions). When all questions for a topic have been processed, a topic-attempt summary is displayed for the user. Subsequently, control is returned to the instructional type index for that specific drill.

Any topic in any unit may be reached by choosing appropriately at each level of index. Alternatively, the user may choose to terminate the drill session in any of the indexes by choosing that option.

Remedial and Criterion Problem Data Sets

Drill topic problems are of two general types: those that display raw data only and those that display raw data and/or intermediate computations. For drills that require the user to work strictly from the raw data, problem data sets are designated as "criterion" or "remedial." A criterion problem data set consists of a specific minimum number of scores or values. A remedial problem data set is one which consists of fewer than the minimum criterion size. The distinction between remedial and criterion data sets is not made for problems that initially present intermediate computations to the user.

Upon initial entry to a topic, a minimum criterion level problem is presented. If the user makes computational or procedural errors (detected during the response/analysis phase) and if the same topic is immediately attempted again, a new, remedial problem data set is presented. If no errors are made and the same topic is immediately attempted again, a new, larger data set is displayed. Typically there is a difference of three to five values between successive problems.

If a user cycles through the same topic several times without branching to a topic index or the unit index, then increasingly larger or smaller data sets are presented depending on whether the student's answers to the previous topic-attempts were completely correct or in error. If a user branches to a topic index before reattempting a topic, the size of the next presented problem data set is again the initial criterion size for that problem.

Successive remedial problem sets are allowed to decrease in size to a predetermined minimum, after which the number of values presented returns to the initial criterion size. Likewise, criterion data sets may increase in size to a preset maximum.

System Pauses

At appropriate places, the system pauses to allow the user to read a passage (case 1), enter a calculated answer (case 2), or make a branching selection (case 3). In case 1, the message:

PRESS 'RETURN'

is displayed. To continue, the user need only press the RETURN or ENTER key. In case 2, the message:

YOUR ANSWER =

is displayed, or a simple line space occurs following a question. To continue, the user must enter a numeric or character answer (as appropriate). Entry of wrong data type, insufficient data, or excess data results in HP-2000 warning messages. The user should follow system instructions and, if necessary, enter the correct data. In case 3, the DPST System expects the user to enter a numeric constant indicating the selected choice from a currently displayed branching options list. Failure to enter one of the specified numbers may result in program termination and automatic log off.

Timing-Out

If the user fails to press the RETURN key after a reading pause (case 1), enter an answer after a question (case 2), or enter a proper numeric constant after an options list (case 3), the DPST System begins a timing-out sequence. In case 1, after 255 seconds, the bell rings several times and the message:

NEED MORE TIME? (YES OR NO)

is displayed. If the user fails to respond within 60 seconds to this question, the program continues normally and displays the next passage of text or question. If the user desires more time before continuing, he enters YES or Y and is allowed an additional 255-second period. This cycle repeats indefinitely. To continue immediately, the user enters NO or simply presses RETURN. The system proceeds normally with the next passage or question. Sometimes, because of the complexity of a computation or lack of understanding of an algorithm, a student might conceivably repeat the pause cycle several times before continuing normally.

In case 2, after 255 seconds, the bell rings and the timing-out message is displayed as above. If the user fails to respond within sixty seconds to the timing-out question, a zero is entered as the student's response to the original numeric question and the program continues normally. (A zero would typically be an incorrect answer.) If the user enters anything (or simply presses RETURN) to the timing-out message, the system pauses for an additional 255 seconds. To continue normally, the user must enter a numeric constant in response to the original numeric question. Character answers (usually YES or NO or Y or N) are time-out as wrong without any message or additional time.

In case 3, the required response is a numeric constant corresponding to an instructional option displayed in a list. Failure to enter a unit number after the unit index or an instructional type number after an instructional type index within 60 seconds results in automatic log off. Failure to respond to a topic index within 255 seconds displays the timing-out message. Failure to respond within the additional 60 seconds also results in automatic log off. Whenever an automatic log off occurs, the final performance file update of DPSPF1 occurs (if appropriate). In cases 2 and 3, pressing the RETURN key without a numeric constant causes the system to act as though the user had timed-out at that point.

A small number of questions in units 1 and 2 require the user to enter a series of numbers on one line. The MAT INPUT statement is used and no timing-out sequence is initiated for failure to respond.

Logging Off and Final Performance File Update

If the user responds to all prompts, the system continues to display descriptive material, problems, questions, and results indefinitely. After each topic-attempt, an option to log off is displayed. Choice of this option updates the performance file DPSPF1 (for required users) and automatically logs the user off the terminal. Optionally, the user may press the BREAK key to terminate abnormally at any time, since this key is not disabled. However, abnormal termination of the program prevents the final performance file update; credit for the drill session may be lost.

Performance Files Operation

The DPST System is capable of storing certain information about the performance of required users for whom performance files have been created. This information consists of the number of times each topic was attempted and attempted correctly, the length of time spent on the system, a listing of attempted topics in the order they were attempted, the percentage of correct answers on each attempt, the time spent on each topic-attempt, and an indication of whether each problem set was remedial or criterion. These data are available in two standard report forms which are generated by available support programs executed by the instructor.

Two performance files, DPSPF1 and DPSPF2, must be created and initialized in the group account corresponding to a class of users. To create the files, the instructor must log onto the group account and enter:

```
CRE-DPSPF1,n
MWA-DPSPF1
UNR-DPSPF1
```

and

```
CRE-DPSPF2,n
MWA-DPSPF2
UNR-DPSPF2
```

where "n" refers to the maximum number of expected required users in the group account. When requesting a group account, the instructor must request that 2n blocks of storage be allocated to the account. Individual accounts need not have any library allocation.

DPSPF1 and DPSPF2 are initialized by a support program, CREFLS. The instructor must enter:

`EXE-CREFLS.A900`

Prompts introduce the purpose of the support program and specify all necessary input. All or part of the two performance files may be initialized at any time as students are added to or dropped from the class rolls. Thus, the same Student Record Number may be reused without difficulty.

The instructor should assign one (integer) Student Record Number (1,2,3,...,n) to each student. To initialize the performance files, the instructor must reference this number. Additionally, a valid nine-digit Social Security Number (no embedded blanks or other characters) and the first and last names of each student must be entered, with a maximum of 15 characters each for the names.

Identification data must be exactly correct. If the identification data that the required user enters at the terminal and the corresponding data listed in the DPST files do not match, the student is not allowed access to the system as a required user.

Generating Performance Files Summaries

Two reports are available, Performance Report and Research Report. Both are designed to provide the instructor with basic information about required user performance on the DPST System. Both reports are produced on a hard-copy terminal by supplied support programs. The progress of one or many students may be requested at any time.

The Performance Report (see Appendix C) is essentially a tally of the number of times a user has (a) attempted each topic and (b) answered all questions for each topic correctly using a criterion level data set. Topics not attempted are not listed in the report. In addition, the total number of minutes spent on the DPST System is reported.

To generate a Performance Report, the instructor utilizes the support program PRSPF1. He must log onto the group account corresponding to the class of users and enter:

```
EXE-PRSPF1.A900
```

Prompts explain the purpose of the support program and indicate the required input. The instructor should respond to each prompt appropriately by entering (a) the class name or number, (b) the instructor's name, (c) the Student Record Number of the first student to be reported, and (d) the Student Record Number of the last student to be reported. At this point, PRSPF1 displays the message:

```
ROLL PAPER FORWARD FOR NEAT COPY. PRESS 'RETURN'
```

and pauses for 255 seconds or until the RETURN key is pressed. After one report is generated, the instructor is given an opportunity to generate additional reports.

The Research Report (see Appendix D) is a listing of the topics attempted in the order each user tried them. For each topic-attempt, the number of minutes required and the percentage of correct answers is also reported. If zero minutes are indicated, then less than one minute of time was required for the attempt. If the percentage of correct answers is negative, then a remedial data set was attempted.

To generate a Research Report, the instructor utilizes the support program PRSPF2. He should log onto the group account and enter:

```
EXE-PRSPF2.A900
```

Prompts explain the purpose of the support program and indicate the required input. The instructor should respond to each prompt appropriately by entering the data requested. At this point, the program displays a message and pauses for 255 seconds to allow the instructor to roll the terminal paper forward for a clean copy.

After the report is generated, the instructor is given an option to erase selected topic-attempt summaries. Prompts are given to specify the Student Record Numbers of those students whose records are to be erased. After the files are cleared, the instructor is given an option to generate additional reports.

The file DPSPF2 from which Research Report is generated holds approximately 45 topic-attempt summaries for each student. Therefore, periodically, the interested instructor should copy the file using PRSPF2 and erase the currently listed topic-attempt summaries. Use of the summary erasure option in PRSPF2 does not purge user identification data nor does it affect the companion file DPSPF1. DPSPF2 must not be reinitialized by CREFLS or DPSPF1 will be purged.

If a required user makes more than 45 topic-attempts before the file is copied and erased, the excess attempts are not recorded on DPSPF2. The results are, however, noted normally on DPSPF1 which is formatted to record data continuously throughout the semester. When DPSPF2 becomes full, the user is not notified of the condition. He may still use the system although DPSPF2 will not record his performance.

Organizational Structure and Coding Conventions

The DPST System is composed of 12 major CAI programs, two student performance files, three tables, and four support programs. Where appropriate, functions which are potentially useful in several units or topics are localized in a subroutine library, DPSUBS. Use of these subroutines and the coding conventions outlined below should significantly ease the authoring difficulties of additional topics. Each component of the system is discussed separately.

DPST00 Index Program

The CAI materials in programs DPST01 through DPST11 occupy the majority of space required by the system. DPST00 is the index program which is initially executed by a user at every drill session. The general functions of DPST00 are to (a) initialize common variables, (b) record the duration of the drill session, (c) introduce the general operating details of the DPST System (optional), (d) verify identification data for required users, (e) list the available statistics units (unit index), (f) chain to the selected drill program, (g) update DPSPF1, and (h) log the user off the terminal automatically. The remaining drill programs chain back to DPST00 to (a) display the unit index (chain to line 600) or (b) update DPSPF1 and terminate the drill session (chain to line 1200).

DPST01 Through DPST11 Drill Programs

All eleven CAI drill programs are organized identically. Table 1 summarizes the line location or number conventions which are observed. Strict adherence to these conventions permits the author of a new unit to append the subroutine library to his code. Subroutines which are not needed during the execution of a unit may be conveniently deleted from the workspace before a unit program is saved.

Each drill program consists of several distinct modules. The main program, located below line 500, serves to display the topic index, process a user topic selection, branch to the selected drill, and chain to the index program for the unit index or to terminate the session. The drills for a specific unit are contained as instructional subroutines beginning at lines 500, 1000, 1500, and so forth for as many as are needed. The coding method for each instructional subroutine is idiosyncratic to the topic.

In a few units, special purpose subroutines serve a function useful only to that unit. These unit-specific subroutines are located between lines 4000 and 4999.

Table 1

Line Number Conventions for Major Functions of Drill Programs

Line	Function
10	Dimension common variables
.	Dimension and initialize local variables
.	Open and assign files
.	Display topic index and process selection
.	Call selected topic (instructional subroutine)
.	Branch to topic index or Chain to DPST00
500	Instructional subroutine #1
.	Initialize local variables
.	Display instructional type index and process selection
.	Display didactic material
.	Present problem data set and instructions
.	Present questions, process answers
.	Display topic-attempt summary
999	Return to main program
1000	Instructional subroutine #2
1500	Instructional subroutine #3
.	
.	
4000	Unit-specific subroutines
5000	Computational and formatting subroutines
9000	Formula printing subroutines
9999	End

Performance File DPSPF1

DPSPF1 is a BASIC formatted file with a default record length of 256 words. Access is direct by Student Record Number (1,2,3,...,n). DPSPF1 must be created before any performance data can be collected. The file is created by HP-2000 system commands and is initialized by the support program CREFLS. During CAI drill execution, DPSPF1 is updated only in DPST00. The support program PRSPF1 accesses the file to generate the Performance Report. Each record in DPSPF1 is formatted as shown in Table 2.

Table 2

Student Record Format in File DPSPF1

Position	Information	Data Type	Variable Convention
1	Social Security Number	Character (9)	N$
2	Last name	Character (15)	N1$
3	First name	Character (15)	NO$
4	Time on system	Numeric scalar	F1
5	Topic-attempt summary array	Numeric array (55,2)	F(*,*)

Subscript one in F(*,*) references the topic number. Subscript two references topic-attempt data. F(*,1) refers to the number of topic-attempts while F(*,2) refers to the number of criterion level topic attempts with a score of 100%. The variables NO$ and F(*,*) are carried in common.

Performance File DPSPF2

DPSPF2 is a BASIC formatted file with a default record length of 256 words. Access is direct by Student Record Number (1,2,3,...,n). DPSPF2 must be created before any performance data can be collected. The file is created by HP-2000 system commands and is initialized by the support program CREFLS. Access and update of the file occur immediately following each topic-attempt in Subroutine 6800. The name and location (group account) of DPSPF2 are carried in common by F$(11). The support program PRSPF2 accesses the file to generate the Research Report. Each record in the file is formatted as shown in Table 3.

Table 3

Student Record Format in File DPSPF2

Position	Information	Data Type	Variable Convention
1	Social Security Number	Character (9)	N$
2	Last name	Character (15)	N1$
3	First name	Character (15)	N0$
4	Number of topic-attempt summaries in record	Numeric scalar	F1
5	Topic-attempt summary	Character (8)	R$
6	"	"	"
.	.	.	.
.	.	.	.

A topic-attempt summary written on the file DPSPF2 is an eight-character code which contains sufficient information for the support program PRSPF2 to recreate the topic-attempt summary including (a) the name of the topic, (b) the length of time in minutes required for the topic-attempt, (c) the percentage of correct answers for the attempt, and (d) an indication of whether the problem data set was of criterion or remedial size. Encoding of R$ occurs in Subroutine 6800 immediately after an attempt is completed. Each topic-attempt summary stored in DPSPF2 is formatted as shown in Table 4.

Table 4

Format of Topic-Attempt Summary Variable R$

Position	Information
1-2	Topic number (1-38 currently used)
3-4	Time on topic in minutes
5-8	Optionally signed percentage of correctly answered topic questions (negative sign indicates topic-attempt was on a remedial data set)

Up to 45 topic-attempt summaries can be stored in each record. If a record is full when an update is attempted, no update occurs. Failure to update the file is transparent to the user. To prevent loss of information the instructor should periodically copy DPSPF2 and erase all topic-attempt summaries.

Chi-square Table DPX2F

DPX2F is a one record (256 word) BASIC formatted file containing 90 numeric chi-square values (*df* = 1-30 for *p* = .05, .01, .001). Access is sequential. The following code returns a specific chi-square value:

```
READ #1,1
ADVANCE #1; 3*(R-1)+(C-1),X
READ #1; X2
```

where DPX2F has been previously assigned as file #1. Here, *R* specifies the required degrees of freedom, *C* specifies the required *p* level, X is the return variable required by the BASIC language, and X2 is the accessed chi-square value. Table DPX2F is accessed in CAI topics requiring users to enter a chi-square value as a response to a question. Table 5 shows the format of DPX2F.

Table 5

Format of Chi-square Table DPX2F

Position	Chi-square Value	Data Type
1	*df* = 1, *p* = .05	Numeric scalar
2	*df* = 1, *p* = .01	"
3	*df* = 1, *p* = .001	"
4	*df* = 2, *p* = .05	"
5	*df* = 2, *p* = .01	"
.		
.		
90	*df* = 30, *p* = .001	Numeric scalar

T Table DPTF

DPTF is a one record (256 word) BASIC formatted file which contains 90 numeric *t* values (*df* = 1-30 for *p* = .05, .01, .001). Access is sequential. The following code returns a specific *t* value:

```
READ #1,1
ADVANCE #1; 3*(R-1)+(C-1),X
READ #1; T
```

where DPTF has been previously assigned as file #1. Here, *R* specifies the required degrees of freedom, *C* specifies the required *p* level, X is the return variable required by the BASIC language, and *T* is the accessed *t* value. Table DPTF is accessed in CAI topics requiring the user to enter a *t* value in response to a question. Table 6 shows the format of DPTF.

Table 6

Format of *T* Table DPTF

Position	*t* Value	Data Type
1	*df* = 1, *p* = .05	Numeric scalar
2	*df* = 1, *p* = .01	"
3	*df* = 1, *p* = .001	"
4	*df* = 2, *p* = .05	"
5	*df* = 2, *p* = .01	"
.		
.		
.		
90	*df* = 30, *p* = .001	Numeric scalar

Topic Title Table DPTOPS

DPTOPS is a three record (256 words per record) BASIC formatted file which contains character representations of the titles of currently available CAI topics. Approximately 47 titles can be accommodated before the file must be expanded. Access to DPTOPS is sequential. The following code will access a specific title:

```
READ #1,1
ADVANCE #1; N,R
READ #1; T$
```

where DPTOPS has been previously assigned as file #1. Here, N specifies the topic number (see Appendix A for topic titles and numbers), R specifies the return variable required by the language, and T$ specifies the accessed topic title. Titles are accessed only by PRSPF1 and PRSPF2. Table 7 shows the format of DPTOPS.

Table 7

Format of Topic Title Table DPTOPS

Position	Information	Data Type
1	Number of topic titles	Numeric scalar
2	Title for topic #1	Character (30)
3	Title for topic #2	"
.		
.		
.		
39	Title for topic #38	Character (30)

File Initialization Program CREFLS

CREFLS is a short program which initializes the performance files DPSPF1 and DPSPF2 in a group account. If these files do not exist, a message is generated requesting that the instructor create these files. Otherwise, prompts specify the required input. The instructor should carefully record student identification data and corresponding Student Record Numbers. To check the accuracy of identification data, execute either support program PRSPF1 or PRSPF2. If erroneous identification data has been entered, reexecute CREFLS and reinitialize only those records containing errors.

The program CREFLS directly accesses DPSPF1 and DPSPF2 and writes identification data onto the proper records (as specified by the Student Record Numbers). In addition, the performance array F on DPSPF1 is zeroed for each new student. The topic-attempt counter F1 on DPSPF2 is set to zero.

One or more records may be initialized or reinitialized during the course of the semester without creating or purging the files themselves. No provision exists for expanding the performance files during the semester. Therefore, adequate space must be reserved at the outset to accommodate all potential users. The library space in the group account should be 2n or greater where n is the number of required users. DPSPF1 and DPSPF2 are not initialized or accessed for casual users.

Topic Title Modification Program CRETOP

CRETOP is a short program designed for use by future authors who wish to extend the system or change topic titles. Great care should be exercised in utilizing the program since erasure of part or all of DPTOPS can occur.

The program prompts the author who may (a) add topic titles to DPTOPS, (b) change topic titles, or (c) print or display the entire topic list. All topic titles are carried in DPTOPS as character strings of length 30 (29 characters followed by a period). It is advisable to generate a copy of the current list before and after changes are made to DPTOPS to check the accuracy of the alteration or addition. Since CRETOP and DPTOPS are located in the system library, CRETOP can be used only with the cooperation of the Hewlett-Packard system manager.

Performance Report Generating Programs PRSPF1 and PRSPF2

Programs PRSPF1 and PRSPF2 are short programs which access and report the contents of DPSPF1 and DPSPF2 respectively. Reports on one or more required users may be produced by following the program prompts. Users must execute both programs from the group account which contains the performance files. It is recommended that these reports be produced on a hard-copy terminal for permanent records or off-line review. Each report is dated and marked with the current time when it is produced.

Data Structures and Variable Conventions

Standard data structures and variable naming conventions are used in programs DPST00 through DPST11 to aid in the design, initial coding, debugging, and maintenance of the DPST System. Minor variations in the overall scheme (such as smaller data vectors or nonstandard local variables) occur occasionally in the CAI drill programs where required or permitted by local conditions. All reusable routines in the subroutine library DPSUBS adhere strictly to the conventions described in the following sections.

All array and character variables are initially dimensioned to their maximum size in the COMMON statement or dimension statement at the beginning of each main program in the system. Vectors may be redimensioned within the instructional and computational subroutines, so care must be taken to set vector sizes as needed when using the subroutine library.

Scalar variable names (single values) consist of a single letter optionally followed by a single digit. Vectors are distinguished from scalars by the presence of subscripts. All references to specific vectors are either of the general form, V(*), where "*" refers to all elements of the vector, or of the form, V(110) or V(V0), where "110" and "V0" specify the current active dimension of the corresponding vector. All character variables are identified by the "$" in the reference such as "A$(5)" where "5" is the number of letters.

Standard DPST System Variables

Certain variable names are reserved for specific purposes by the DPST System upon initial execution of DPST00 and are passed in common between programs. Variable conventions are delineated in Table 8.

Table 8

Standard DPST System Variable Conventions

System Variables	Information
F0	Student Record Number
F8	Hour drills begun
F9	Minute drills begun
F(55,2)	Performance array on DPSPF1
F$(11)	Qualified DPSPF1 name (e.g., "DPSPF1.I400")
F0$(3)	Flag specifying whether a user is a required user ("YES" or "NO")
A$(5)	System library designation ("A900")
M$(11)	Qualified index program name ("DPST00.A900")
N0$(15)	User's first name
N1$(15)	User's last name

Since all system variables are passed in common between the index program and the CAI drill program, it is mandatory that additional units begin with an identical COMMON statement. Otherwise, the linkage conventions established here are nullified.

Standard Unit and Topic Variables

Certain variables are reserved for specific purposes by each unit or topic. These variables are reset as required by each CAI unit or topic. Unit and topic variables are not passed between programs but are local to the programs or program segments in which they are active. These variable conventions are delineated in Table 9.

Table 9

Standard Unit and Topic Variable Conventions

Program Variables	Information
Ø$(30)	Topic title
Ø0	Unit number
Ø1	Topic number
Ø2	Instructional type selection
Ø3	Size of the current problem data set
Ø4	Number of correct responses to questions
Ø5	Number of questions in topic
Ø6	Minimum criterion data set size
Ø7	Unused
Ø8	Hour topic started
Ø9	Minute topic started

Note: Here, "Ø" represents the letter "OH."

Standard Local Subroutine Variables

Local variables for subroutines (loop indexes, temporary storage, etc.) have been selected from the following: I, I0-I9, I$(50), J, J0-J9, J$(4), K, K0-K9, L, L0-L9, R, R$(8). The character variables should be dimensioned at the beginning of any new CAI drill unit. In general, it is not safe to assume that any of these variables will retain the same values upon entry and return from any subroutine. Exceptions to this general scheme are noted under the description of specific subroutines.

Standard Numeric Data Structures

Eleven numeric vectors, initially dimensioned at the outset of each major program to 110 elements, are used in the DPST System to store generated values or user input. Subroutines access these vectors for computational and formatting purposes. Table 10 shows the observed conventions.

Each CAI drill unit utilizes a subset of these vectors according to the requirements of the subsumed statistical drills. The common maximum initial vector size is determined from the maximum number of values produced by the population generating Subroutine 5000. If some other method of generating data into the system is used, this restriction on vector size can be discarded. Also, since the actual size of various samples is under control of the programmer, all vectors need not be dimensioned to 110 elements. Exceptions are noted under the descriptions of individual subroutines.

Table 10

Standard Numeric Data Structure Designations

Name and Dimension	Designation
W(110)	Working vector
A(110)	"
B(110)	"
C(110)	"
D(110)	"
E(110)	"
P(1,110)	Horizontal printing vector
V(110)	Population storage vector
X(110)	Sample storage vector
Y(110)	"
Z(110)	"

Standard Simple Statistics Variables

Ten scalar variables are reserved to store ten simple sample statistics computed from the sample values contained in the central working vector W(*). The meanings of these reserved variables are indicated in Table 11.

The vectors X(*), Y(*), and Z(*) are intended to store separate samples if needed for a specific problem. Simple sample statistics for values in these vectors are stored in scalar variables after the pattern described in Table 11. Reserved variables are X0-X9, Y0-Y9, and Z0-Z9. Also, the scalar variables A0, B0, C0, D0, E0, and V0 are reserved to store the number of values currently in the corresponding vectors.

Table 11

Standard Simple Statistics Variable Conventions

Variable Name	Information
W0	The current number of values in W(*)
W1	Sum of values
W2	Sum of squared values
W3	Mean of values
W4	Sum of squared deviation scores
W5	Variance of the sample
W6	Standard deviation of the sample
W7	Estimated population variance
W8	Estimated population standard deviation
W9	Standard error of the mean

Subroutine Library DPSUBS

DPSUBS is a library of reusable subroutines intended to facilitate the authoring of additional statistical drill and practice units and topics. Subroutines generate data into V(*) and W(*), perform computations, transform or rearrange data in a single vector, move data between vectors, display values and other information at the terminal, update DPSPF2, and otherwise assist in the control of program execution. Most subroutines are written to be independent from each other so that subroutines not required for a unit can be conveniently and harmlessly deleted from the workspace. Because DPSUBS resides in the system library, additional useful subroutines may be added to DPSUBS only after consultation with the Hewlett-Packard System manager.

Unless noted, values of externally set variables required by a subroutine are unaffected by the action of that routine. Any standard local variables and any specified non-standard local variables used by a subroutine cannot be assumed to be unchanged by the subroutine. Use of such variables in the main programs or instructional subroutines should be done with caution and only after careful inspection of the subroutine code. Otherwise, unpredictable results may occur.

In the following two sections, all the data formatting, computational, and formula displaying subroutines are listed and briefly described. Descriptions for formatting and computational subroutines include (a) lower and upper line number limits, (b) a statement of the subroutine's function or functions, (c) entry requirements or variables which must be assigned values before the subroutine is called, (d) variables whose return values differ from their entry values, and (e) any known exceptional conditions or additional information which may be useful to programmers using these subroutines. Formula displaying subroutines are listed by line numbers and titles.

Data Formatting and Computational Subroutines

5000 - 5044

Function: Fills vector V(*) with a normally distributed population of integers.

Entry Requirements	Returns
M - mean of population	VO - vector size (approximately 100)
S - standard deviation of the population	

Exceptions: V(*) is dimensioned to 110 elements, filled with integers, and then redimensioned to VO elements. The redimensioning requires temporary storage in W(110).

5060 - 5072

Function: Redimensions and fills W(*) with a random sample of integers from V(*).

Entry Requirements	Returns
WO - vector size (should be less than 100)	W(WO) - data
VO - vector size	
V(*) - population of integers	

Exceptions: This subroutine fills W(*) with a random sample from a normally distributed population of integers. This is the method used to generate most samples in the system.

5200 - 5210

Function: Redimensions W(*) to X0; copies X(*) into W(*).

Entry Requirements	Returns
X0 - vector size	W0 - vector size
X(*) - data	W(*) - data

5220 - 5230

Function: Redimensions W(*) to Y0; copies Y(*) into W(*).

Entry Requirements	Returns
Y0 - vector size	W0 - vector size
Y(*) - data	W(*) - data

5240 - 5250

Function: Redimensions W(*) to Z0; copies Z(*) into W(*).

Entry Requirements	Returns
Z0 - vector size	W0 - vector size
Z(*) - data	W(*) - data

5260 - 5270

Function: Redimensions W(*) to A0; copies A(*) into W(*).

Entry Requirements	Returns
A0 - vector size	W0 - vector size
A(*) - data	W(*) - data

5280 - 5290

Function: Redimensions B(*) to AO; copies A(*) into B(*).

Entry Requirements	Returns
AO - vector size	BO - vector size
A(*) - data	B(*) - data

5300 - 5310

Function: Redimensions W(*) to BO; copies B(*) into W(*).

Entry Requirements	Returns
BO - vector size	WO - vector size
B(*) - data	W(*) - data

5320 - 5330

Function: Redimensions A(*) to BO; copies B(*) into A(*).

Entry Requirements	Returns
BO - vector size	AO - vector size
B(*) - data	A(*) - data

5340 - 5350

Function: Redimensions A(*) to WO; copies W(*) into A(*).

Entry Requirements	Returns
WO vector size	AO - vector size
W(*) - data	A(*) - data

5360 - 5370

Function: Redimensions B(*) to W0; copies W(*) into B(*).

Entry Requirements	Returns
W0 - vector size	B0 - vector size
W(*) - data	B(*) - data

5800 - 5810

Function: Redimensions X(*) to W0; copies W(*) into X(*).

Entry Requirements	Returns
W0 - vector size	X0 - vector size
W(*) - data	X(*) - data

5850 - 5874

Function: Copies sample statistics W0-W9 into X0-X9.

Entry Requirements	Returns
W0-W9	X0-X9

5900 - 5910

Function: Redimensions Y(*) to W0; copies W(*) into Y(*).

Entry Requirements	Returns
W0 - vector size	Y0 - vector size
W(*) - data	Y(*) - data

5950 - 5974

Function: Copies sample statistics W0-W9 into Y0-Y9.

Entry Requirements	Returns
W0-W9	Y0-Y9

6000 - 6010

Function: Redimensions Z(*) to W0; copies W(*) into Z(*).

Entry Requirements	Returns
W0 - vector size	Z0 - vector size
W(*) - data	Z(*) - data

6050 - 6074

Function: Copies sample statistics W0-W9 into Z0-Z9.

Entry Requirements	Returns
W0-W9	Z0-Z9

6100 - 6160

Function: Displays 1-5 vectors of equal size (A(*), B(*), C(*), D(*), E(*)) in 1-5 columns.

Entry Requirements	Returns
A0 - vector size	None
A(*), B(*), C(*), D(*) E(*) - data	
J - number of vectors	

6100 - 6160--continued

Entry Requirements	Returns
J1, (J2-J5) - tab positions	
J$ - element access order	
("UP" means print A(1), A(2),..., A(AO),	
"DOWN" means print A(AO), A(AO-1),..., A(1))	

Exceptions: If one vector is to be displayed, A(*) must be used. If two vectors are to be displayed, A(*) and B(*) must be used, and so forth.

6200 - 6267

Function: Displays 1-6 vectors of equal size (W(*), A(*), B(*), C(*), D(*), E(*)) in 1-6 columns.

Entry Requirements	Returns
WO - vector size	None
W(*), A(*), B(*), C(*), D(*), E(*) - data	
J - number of vectors	
J1, (J2-J6) - tab positions	
J$ - element access order	
("UP" means print W(1), W(2),..., W(WO),	
"DOWN" means print W(WO), W(WO-1),..., W(1))	

Exceptions: If one vector is to be displayed, W(*) must be used. Use W(*) and A(*) for two vectors, and so forth.

6300 - 6338

Function: Displays a histogram, scores, and score frequencies.

Entry Requirements	Returns
B0 - vector size	None
A(*) - unique values	
B(*) - frequencies corresponding to unique values in A(*)	

Exceptions: Individual elements in B(*) must be less that forty. I$(50) must be previously dimensioned.

6350 - 6368

Function: Pauses for 255 seconds; displays topic attempt response summary; calls Subroutine 6800 to update file DPSPF2 and performance array F(55,2); pauses for 255 seconds.

Entry Requirements	Returns
NO$, 00, 01, 03, 04, 05, 06, 0$ (see Standard DPST System Variables, Standard Unit and Topic Variables)	None

Exceptions: I$(50) must be previously dimensioned. This subroutine calls Subroutines 6380 and 6800.

6380 - 6394

Function: Allows student a reading pause of 255 seconds (see System Pauses); displays a timing-out message; pauses sixty seconds; returns control to the calling routine.

Entry Requirements	Returns
None	None

Exceptions: I$(50) must be previously dimensioned.

6400 - 6412

Function: Redimensions W(*) to WO; fills W(*) with J categories of nominal integers (1, 2, 3, ..., J).

Entry Requirements	Returns
WO - vector size	W(*) - nominal data
J - number of nominal categories	

Exceptions: If two categories of data are desired (J = 2), then W(*) is randomly filled with 1's and 2's. If J = 3, then W(*) is filled with 1's, 2's, and 3's. Subroutine 6400 is a method of generating data into W(*) and is thus a substitute for the sequence of Subroutines 5000 and 5060 which generate and sample normally distributed integers. If a CAI unit requires only Subroutine 6400 for generating data, then W(*) may be dimensioned as large or small as desired.

<u>6420 - 6436</u>

Function: Converts unsorted interval data in W(*) to unsorted ordinal data (ranks) in W(*).

Entry Requirements	Returns
WO - vector size	W(*) - unsorted ordinal data corresponding to original data in W(*)
W(*) - unsorted interval data	
AO - vector size	
A(*) - sorted ordinal data derived from W(*)	
B(*) - sorted interval data derived from W(*)	

Exceptions: This subroutine can be used only after A(*) and B(*) have been initialized properly (see Subroutine 6450) and W(*) has been restored with the original, unsorted interval data. A logical sequence of coding would be:

(a) Fill W(*) with interval data (Subroutine 5060)
(b) Copy W(*) into X(*) (Subroutine 5800)
(c) Create A(*) and B(*) (Subroutine 6450)
(d) Copy X(*) into W(*) (Subroutine 5200)
(e) Convert W(*) to ordinal data (Subroutine 6420)

For multiple samples of ordinal data, W(*) could then be stored in Y(*) and the five step procedure outlined above repeated.

6450 - 6494

Function: Converts interval data in W(*) to sorted ordinal data in A(*) and sorted interval data in B(*), W(*); redimensions A(*) to WO.

Entry Requirements	Returns
WO - vector size	AO - vector size
W(*) - unsorted interval data	A(*) - sorted ordinal data
	B(*), W(*) - sorted interval data

Exceptions: Calls Subroutines 7178 and 5360.

6500 - 6530

Function: Initiates timing for drill; displays Instructional Type Index (see Drill Sequencing and Operation); accepts user selection.

Entry Requirements	Returns
00, 01, 0$	02, 08, 09

(See Standard Unit and Topic Variables)

Exceptions: If the user fails to respond within sixty seconds, 02 (instructional type selection) is set to "4." Subsequently the user is automatically logged off the system and terminal.

6550 - 6566

Function: Prints W(*) horizontally at standard HP-2000 tab positions.

Entry Requirements	Returns
W0 - vector size	None
W(*) - data	

Exceptions: The vector W(*) is displayed in rows. Nonstandard local variable P(1,*) is utilized. P(1,110) must be previously dimensioned.

6600 - 6628

Function: Pauses up to 255 seconds; accepts single scalar input from terminal; rings the bell and displays timing-out message; pauses up to 60 seconds; sets user response "P" equal to zero if user times-out.

Entry Requirements	Returns
None	P - user response to a numeric question

Exceptions: I$(50) must be previously dimensioned. The nonstandard local variable P is used to return the user's numeric response to the calling routine. If the timing-out message has been displayed, the user must first respond to that message and then enter his numeric response to the original question.

6800 - 6846

Function: Updates file DPSPF2 and performance array F(55,2).

Entry Requirements	Returns
F(55,2), FO$, FO, 01, 04, 05. 08, 09 (standard system, unit, and topic variables)	Topic-attempt summary on on DPSPF2 Updated F(55,2)

Exceptions: R$(8) and I$(50) must previously be dimensioned. Nonstandard local variable F1 is used in addition to those listed above. This subroutine is called by Subroutine 6350.

7100 - 7130

Function: Computes simple statistics W1-W9 from W(*).

Entry Requirements	Returns
WO - vector size	W1-W9 - standard simple statistics variables
W(*) - data	

Exceptions: If WO = 0 then W1-W9 are set equal to zero. If WO = 1, then W3-W9 are set equal to zero.

7150 - 7170

Function: Converts scores in W(*) to deviation scores in A(*) and z scores in B(*); redimensions A(*) and B(*) to WO.

7150 - 7170--continued

Entry Requirements	Returns
WO - vector size	AO, BO - vector size
W(*) - data	A(*) - deviation scores
	B(*) - z scores
	W1-W9 - (see Standard Simple Statistics Variables)

Exceptions: Calls Subroutine 7100. WO must be greater than one.

7178 - 7198

Function: Sorts W(*) into descending order.

Entry Requirements	Returns
WO - vector size	W(*) - data sorted into descending order
W(*) - unsorted data	

7200 - 7254

Function: Condenses raw data in W(*) into unique scores in A(*) and frequencies in B(*), excluding scores with zero frequency; redimensions A(*) and B(*) to AO.

Entry Requirements	Returns
WO - vector size	W(*) - sorted data
W(*) - unsorted data	AO, BO - vector sizes
	A(*) - sorted unique data
	B(*) - frequencies

7260 - 7290

Function: Condenses raw data in W(*) into unique scores in A(*) and frequencies in B(*), including scores with zero frequency; redimensions A(*) and B(*) to AO.

Entry Requirements	Returns
WO - vector size	W(*) - sorted data
W(*) - unsorted data	AO, BO - vector sizes
	A(*) - sorted unique data
	B(*) - frequencies
	H1 - high value in W(*)
	L1 - low value in W(*)

Exceptions: Calls Subroutines 7178 and 7630. Upon return, A(1) = 1, A(2) = 2, ..., A(AO) = AO. Since AO equals the highest value in W(*), that value must be greater than zero and less than 111 to remain within standard dimension bounds.

7300 - 7332

Function: Converts data in W(*) to sorted unique scores in A(*), corresponding frequencies in B(*), cumulative frequencies in C(*), relative frequencies in D(*), and percentile ranks in E(*); redimensions A(*)-E(*) to AO.

Entry Requirements	Returns
WO - vector size	AO, BO, CO, DO, EO - vector sizes

7300 - 7332--continued

Entry Requirements	Returns
W(*) - data	W(*) - sorted data
	A(*) - sorted unique data
	B(*) - frequencies
	C(*) - cumulative frequencies
	D(*) - relative frequencies
	E(*) - percentile ranks

Exceptions: Calls Subroutine 7620. No data with frequency of zero is stored.

7350 - 7376

Function: Converts data in W(*) to percentile scores in W(*); redimensions W(*) to WO = 99.

Entry Requirements	Returns
WO - vector size	WO = 99
W(*) - data	W(*) - percentiles
	AO-EO, A(*)-E(*) as described under Subroutine 7300

Exceptions: Calls Subroutine 7300. This subroutine requires considerable time to execute (ten to twenty seconds, for example) due to the number of calculations and the nested subroutine calls.

7630 - 7644

Function: Finds the largest and smallest values in W(*).

Entry Requirements	Returns
WO - vector size	H1 - largest value in W(*)
W(*) - data	L1 - smallest value in W(*)

Exceptions: Nonstandard local variable H1 is used.

7650 - 7672

Function: Computes the Pearson Product Moment Correlation between data in W(*) and A(*).

Entry Requirements	Returns
WO - vector size	K - Pearson r
W(*), A(*) - data	A1, A2, W1, W2 - standard simple statistics

Exceptions: Nonstandard local variables A1, A2, W1, and W2 are used. Also, because a significant positive correlation between two vectors is possible only when the two vectors are, to some degree, ordered, the programmer must see to this aspect. One way to partially order vector elements is to set WO lower than the actual vector size of W(*) and then to call Subroutine 7178. The exact reduction in WO for this purpose is determined from the degree of correlation required. Trial and error is required.

7680 - 7694

Function: Redimensions B(*) to WO; fills B(*) with the element by element product of W(*) and A(*).

Entry Requirements	Returns
WO - vector size	BO - vector size
W(*), A(*) - data	B(*) - element by element product of W(*) and A(*)

7800 - 7820

Function: Finds and displays the mode(s) in W(*).

Entry Requirements	Returns
WO - vector size	AO, BO - vector sizes
W(*) - data	A(*) - unique data
	B(*) - frequencies
	H1 - largest value in W(*)
	L1 - smallest value in W(*)

Exceptions: Calls Subroutines 7260, 5300, and 7630.

7830 - 7844

Function: Finds median in W(*).

Entry Requirements	Returns
WO - vector size	K - median
W(*) - data	W(*) - sorted data

Exceptions: Calls Subroutine 7178

7850 - 7866

Function: Fills W(*) with transformed scores.

Entry Requirements	Returns
WO - vector size	AO, BO - vector sizes
W(*) - data	A(*) - deviation scores
M - new mean	B(*) - z scores
S - new standard deviation	W(*) - transformed scores

Exceptions: Calls Subroutine 7150. Non-standard local variables M and S are used.

Formula Displaying Subroutines

Lines 9000 to 9998 are reserved for a series of subroutines which display statistical formulas. Each subroutine displays one formula after skipping one line. Symbols used in the formulas are not defined by the subroutines but are standard to the DPST System (see Appendix E for symbol conventions). Table 12 lists the formulas in DPSUBS.

Table 12

Formula Displaying Subroutines

Lines	Formula
9004 - 9012	*Z* score
9014 - 9022	*Z* test with known population mean
9024 - 9032	Standard error of the mean
9034 - 9042	*Z* test for independent samples

Table 12--Continued

Lines	Formula
9044 - 9054	Standard error of the difference between independent means
9056 - 9064	Z test for correlated means
9066 - 9076	Standard error of the difference between correlated means
9078 - 9086	T test with known population mean
9088 - 9096	T test for independent samples
9098 - 9106	T test for correlated samples
9108 - 9116	Raw score "sum of squares"
9118 - 9126	Raw score "sum of products"
9128 - 9138	Standard deviation
9140 - 9148	Variance
9150 - 9160	Estimated standard deviation (S)
9162 - 9170	Estimated variance (S^2)
9172 - 9180	Pearson product moment correlation (r)
9182 - 9192	T test of significance of r
9194 - 9202	Raw score regression equation
9204 - 9210	Standard error of estimate
9212 - 9220	Percentile rank
9222 - 9230	Percentile score
9232 - 9240	Spearman rank correlation (rho)
9242 - 9264	Fourfold point correlation (phi)
9266 - 9274	Point biserial correlation (r_{pb})
9276 - 9284	Partial correlation
9286 - 9300	Chi-square
9302 - 9312	Contingency coefficient

Appendix A

Table 13

Currently Available Statistics Drill Topics

Unit	Topic	Topic Title
1	1	Summation Notation
	2	Arranging Data
	3	Tabled Values
2	4	Grouping Data
	5	Relative, Cumulative Frequency
	6	Percentile Rank
	7	Percentile Score
3	8	Mean
	9	Mode
	10	Median
4	11	Range
	12	Deviation, Mean Deviation
	13	Sum of Squares, Products
	14	Sample Standard Deviation, Variance
	15	Population Standard Deviation, Variance
5	16	Standard Scores
	17	Transformed Scores
	18	Confidence Interval, Mean
6	19	*Z* Test, Known Population Mean
	20	*Z* Test, Unrelated Samples

Table 13--Continued

Unit	Topic	Topic Title
	21	_Z_ Test, Related Samples
7	22	_T_ Test, Known Population Mean
	23	_T_ Test, Unrelated Samples
	24	_T_ Test, Related Samples
8	25	Pearson Product Moment Correlation
	26	Significance of _R_
	27	Raw Score Regression Equation
	28	Standard Error of Estimate
9	29	Spearman Rank Correlation
	30	Fourfold Point Correlation
	31	Point Biserial Correlation
	32	Partial Correlation
10	33	Chi-square, A Priori Cell Frequencies
	34	Goodness of Fit
	35	Contingency Table
	36	Significance of Changes
	37	Contingency Coefficient
11	38	One Way Analysis of Variance

Appendix B

Instructions for Logging On and Off Interactive Terminals

1. Turn on power switch at terminal.
2. Press RETURN key.
3. Press LINEFEED key.
4. The message, PLEASE L0G 0N, should appear. If you do not receive this message, wait a few seconds for the terminal to warm up before repeating steps two and three. If the terminal still fails to respond, turn the terminal off and check with the computing center.
5. Type HELL0- followed by your four-character ID code and a comma. Example: HELL0-I400,
6. Hold down the control key (CNTL) and type your six-digit password. Example: HELL0-I400,dddddd
   The password should <u>not</u> print on the terminal.
7. Press the RETURN key.
8. Several messages will be displayed. When the single word, READY, appears, type: EXE-DPST00.A900
9. Follow the prompts given by the program. If you wish to terminate the drill session, choose that option in any of the indexes displayed by the program. If you must terminate the drill session immediately, press the BREAK key and type BYE followed by pressing the RETURN key. Terminating the drill session in this manner will prevent your receiving credit for the session if you are a required user.

Appendix C

Example of the Performance Report

```
-----------------------------------------------
PERFORMANCE REPORT (DPSPF1): STATISTICS D & P
-----------------------------------------------


**********
PSYCHOLOGY 570 DR. JONES        HOUR: 19    DATE:  228   78
**********


STUDENT# 1   555555555   SMITH, JOE            TIME= 35 MINS.

TOPIC#  TOPIC TITLE                  ATTEMPTS     100% AT CRITERION

 1      SUMMATION NOTATION              1               1
 2      ARRANGING DATA                  3               1
 3      TABLED VALUES                   1               0
 4      MEAN                            4               2
 5      MODES                           2               0


                      END OF REPORT

ROLL PAPER FORWARD.  PRESS 'RETURN.'
```

Appendix D

Example of the Research Report

```
        ---------------------------------------
        RESEARCH REPORT (DPSPF2): STATISTICS D & P
        ---------------------------------------

NOTE: NEGATIVE NUMBERS INDICATE REMEDIAL DATA SETS.


**********
PSYCHOLOGY 570 DR. JONES      HOUR: 20      DATE: 228   78
**********


STUDENT# 6    999999999   JONES, PETER

NO.       TOPIC# TOPIC TITLE                    MIN   % COR

 1          1    SUMMATION NOTATION              10    10
 2          9    MODE                             2    66
 3          9    MODE                             5    -100
 4          9    MODE                             4    100
 5          8    MEAN                             1    0
 6         16    STANDARD SCORES                  3    33
 7         30    FOURFOLD POINT CORRELATION       2    50

                   END OF REPORT

ROLL PAPER FORWARD.  PRESS 'RETURN.'
```

Appendix E

Special Statistical Symbol Conventions

Throughout the DPST System, a wide range of symbols is employed in the CAI textual presentations to represent numerical quantities. The didactic segments of each topic fully describe the meaning of all symbols. Most notation is readily interpretable by the user (e.g., *M* = mean; *DF* = degrees of freedom; *Y*' = predicted y value; M(SAMPLE) = mean of a sample). The more difficult notation used in the system is defined in Table 14.

Table 14

Special Statistical Symbol Definitions

Symbol	Definition
*	Signifies a multiplication
X.SQ	Square of a value
SUM(X)	Sum of all x values in a sample
SUM(X.SQ)	Sum of all x-squared values
(SUM(X)).SQ	Square of the quantity "SUM(X)"
SUM(XY)	Sum of all the "x * y" products
(SUM(XY)).SQ	Square of the quantity "SUM(XY)
SS(X)	Sum of the squared deviation scores of x ("sum of squares")
SP(XY)	Sum of the products of the deviation scores of x and y ("sum of products")
SE(MEAN)	Standard error of the mean

Table 14--Continued

Symbol	Definition
SE(MEANDIFF)	Standard error of the difference between two independent means
SE(CORMEANDIFF)	Standard error of the difference between two correlated means
SE(EST)	Standard error of the estimate
SQRT(X)	Square root of x

References

Abranovic, W., Ageloff, R., & Frederick, D. Time-sharing computer systems as a teaching tool. The American Statistician, 1972, 26(1), 34-38.

Alpert, D., & Bitzer, D. L. Advances in computer-based education. Science, 1970, 167, 1582-1590.

Anastasio, E. J. Computer-based education: Obstacles to its use and plans for future action. Viewpoints, 1974, 50(4), 11-37.

Anderson, T. H., Standiford, S. N., & Alessi, S. M. Computer assisted problem solving in an introductory statistics course. (Center for the Study of Reading Tech. Rep. No. 56). Champaign, Illinois: University of Illinois at Urbana-Champaign, August 1977. (ERIC Document Reproduction Service No. ED 146 563)

Appelbaum, M. I., & Guthrie, D. Use of computers in undergraduate statistics instruction. Proceedings of a conference on computers in the undergraduate curricula. Iowa City: University of Iowa, 1970.

Ashburn, A. G. Use of the computer to increase individualized effort. Improving College and University Teaching, 1977, 25, 57-58.

Atkinson, R. C. Computerized instruction and the learning process. American Psychologist, 1968, 23, 225-239.

Atkinson, R. C., & Wilson, H. A. Computer-assisted instruction. Science, 1968, 162, 73-77.

Bailey, D. E., & Polson, P. G. Real-time computing in psychology at the University of Colorado. American Psychologist, 1975, 30, 212-218.

Baker, F. B. Computer-based instructional management systems: A first look. Review of Educational Research, 1971, 41(1), 51-70.

Brudner, H. J. Computer-managed instruction. Science, 1968, 162, 970-976.

Bruell, J. H. Completely computer-managed selfpaced classes. AEDS Journal, 1976, 9(3), 54-59.

Cooley, W. M. Computer assisted instruction in statistics. Paper prepared for the Conference on Statistical Computation, University of Wisconsin Computing Center, April 30, 1969. (ERIC Document Reproduction Service No. ED 035 294)

Counts, G. The creation and solution of unique statistical problems for each student. The Journal of Experimental Education, 1969, 37(3), 17-20.

Cunningham, G. K. Effect of computer-based instruction (CBI) frame style on performance in statistics. (Doctoral dissertation, University of Arizona, 1975). Dissertation Abstracts International, 1976, 36, 5142-A.

Diegert, C. Some experience with interactive computing in teaching introductory statistics. Paper presented at the Shared Educational Computing System 2nd Annual Meeting and Conference, Ithica, New York, May 2, 1974. (ERIC Document Reproduction Service No. ED 100 366)

Dowsey, M. W. Easy author-entry systems: A review of a prototype. International Journal of Man-Machine Studies, 1974, 6, 401-419.

Dwyer, T. A. Some principles for the human use of computers in education. International Journal of Man-Machine Studies, 1971, 3, 219-239.

Dwyer, T. A. Teacher/student authored CAI using the NEWBASIC system. Communications of the ACM, 1972, 15(1), 21-28.

Dwyer, T. A. Heuristic strategies for using computers to enrich education. International Journal of Man-Machine Studies, 1974, 6, 137-154.

Edgar, S. M. Teaching statistics, while simultaneously saving time, chalk, etc.... Paper presented at the Conference on Computers in the Undergraduate Curricula, The Claremont Colleges, Claremont, California, June 18-20, 1973. (ERIC Document Reproduction Service No. ED 079 993)

Edwards, J., Norton, S., Taylor, S., Van Dusseldorp, R., & Weiss, M. Is CAI effective? AEDS Journal, 1974, 7(4), 122-126.

Ellsworth, D. W. Easy to hard sequence of computer simulated psychological experiments. In T. C. Willoughby (Ed.), Proceedings of 1976 conference on computers in the undergraduate curricula. Binghamton, N. Y.: State University of New York at Binghamton, 1976.

Faust, G. W. Design strategy and the TICCIT system. Viewpoints, 1974, 50(4), 91-101.

Finch, J. M. An overview of computer-managed instruction. Educational Technology, 1972, 12(7), 46-47.

Finch, J. M. Computer managed instruction: An annotated bibliography. Audiovisual Instruction, 1972, 17(3), 72-76.

Forsythe, A. B., & Bleich, D. Some experience using computers to assist in the teaching of basic statistics. International Statistical Review, 1973, 11, 245-251.

Frederick, F. J. PLANIT: A 'machine-independent' CAI language and system. Viewpoints, 1974, 50(4), 81-90.

Frye, C. H. CAI languages: Their capabilities and applications. In R. C. Atkinson & H. A. Wilson (Eds.), Computer-assisted instruction. New York: Academic Press, 1969.

Garrett, H. G. SAMDS: A program to generate empirical sampling distributions of the mean. Proceedings of a conference on computers in the undergraduate curricula. Iowa City: University of Iowa, 1970.

Grubb, R. E. Learner-controlled statistics. In R. C. Atkinson & H. A. Wilson (Eds.), Computer-assisted instruction. New York: Academic Press, 1969.

Gulliford, N. L., & Blau, T. H. Instructional technology: Automated educational systems and instructional devices. JSAS Catalog of Selected Documents in Psychology, 1976, 6, 121. (Ms. No. 1384)

Hall, K. A. CARE: Computer assisted renewal education. Viewpoints, 1974, 50(4), 65-79.

Hewlett-Packard Company. HP 2000 access instructional dialogue facility: Author's manual. Cupertino, California: Author, 1975.

Hewlett-Packard Company. HP 2000 BASIC reference manual. Santa Clara, California: Author, 1976.

Howze, G. R. The use of computer time-sharing to individualize instruction in a course of elementary statistics. Social Science Information, 1973, 12(6), 149-157.

Jamison, D., Suppes, P., & Wells, S. The effectiveness of alternative instructional media: A survey. Review of Educational Research, 1974, 44(1), 1-67.

Kearsley, G. P. Some 'facts' about CAI: Trends 1970-1976. Journal of Educational Data Processing, 1976, 13(3), 1-12.

Kelly, J. T., & Anandam, K. Response system with variable prescriptions (RSVP): A faculty-computer partnership for enhancement of individualized instruction. Paper presented at the SCUP/M-DCC/CAUSE Workshop on Advance Planning and Management Systems in Education, Miami Beach, Florida, March, May 1976.

King, A. T. Impact of computer-based instruction on attitudes of students and instructors: A review. JSAS Catalog of Selected Documents in Psychology, 1975, 6, 21. (Ms. No. 1205)

Knief, L. M., & Cunningham, G. K. Effects of tutorial CBI on performance in statistics. AEDS Journal, 1976, 9(2), 43-45.

Koh, Y. O. The computer as an instructional tool for the statistics courses. Proceedings of a conference on computers in the undergraduate curricula. Iowa City: University of Iowa, 1970.

Kurshan, B. L. Current state of the art in computer-assisted instruction: An annotated bibliography. AEDS Journal, 1974, 7(3), 80-91.

Lekan, H. A. Index to computer assisted instruction (2nd ed.) Boston: Sterling Institute, 1970.

Lekan, H. A. Index to computer assisted instruction (3rd ed.). New York: Harcourt Brace Jovanovich, Inc., 1971.

Lippert, H. T. Computer support of instruction and student services in a college or university. Educational Technology, 1971, 11(5), 40-43.

Lippey, G. Computer managed instruction: Some strategic considerations. Educational Technology, 1975, 15(1), 9-13.

McLean, L. D. It's almost time for CAI. Interchange, 1973, 4(4), 35-47.

Miller, J. Understanding the total process. Change, 1976, 8(2), 46-49.

Moore, C. N. Computer assisted laboratory problems for teaching business and economic statistics. Paper presented at the Conference on Computers in the Undergraduate Curricula, The Claremont Colleges, Claremont, California, June 18-20, 1973. (ERIC Document Reproduction Service No. ED 081 221)

Paloian, A. Y. An interrogative authoring system. International Journal of Man-Machine Studies, 1974, 6, 421-444.

PLATO IV Software Group. PLATO IV authoring. International

Journal of Man-Machine Studies, 1974, 6, 445-463.

Rubner, V. P., Behr, G. E., & Baker, F. B. STAT*CONCEPT: An interactive computer package supporting a first course in educational statistics. Paper presented at the annual meeting of the American Educational Research Association, Chicago, Illinois, April 1974. (ERIC Document Reproduction Service No. ED 090 036)

Sherr, L. A., Tuggle, F. D., & Fitch, H. G. Design and tests of computer-furthered instruction. International Journal of Man-Machine Studies, 1974, 6, 105-124.

Sidowski, J. B. Instrumentation and computer technology: Applications and influences in modern psychology. American Psychologist, 1975, 30, 191-196.

Skinner, B. F. The science of learning and the art of teaching. Harvard Educational Review, 1954, 24(2), 86-97.

Skinner, B. F. Teaching machines. Science, 1958, 128, 969-977.

Skinner, B. F. The technology of teaching. New York: Appleton-Century-Crofts, 1968.

Snyder, R. E. Computer simulations in teaching psychology. Paper presented at the annual meeting of the American Educational Research Association, New York City, April 4-8, 1977. (ERIC Document Reproduction Service No. ED 143 313)

Spelt, P. F., & Schafer, S. R. The use and evaluation of EXPER SIM: What does it teach and how should it be evaluated? In T. C. Willoughby (Ed.), Proceedings of the 1976 conference on computers in the undergraduate curricula.

Binghamton, N. Y.: State University of New York at Binghamton, 1976.

Spuck, D. W., Hunter, S. N., Owen, S. P., & Belt, S. L. Computer management of individualized instruction. Madison, Wisconsin: Wisconsin Research and Development Center for Cognitive Learning, 1975.

Suppes, P., Jerman, M., & Brian, D. Computer assisted instruction: Stanford's 1965-1966 arithmetic program. New York: Academic Press, 1968.

Suppes, P., & Morningstar, M. Computer-assisted instruction at Stanford, 1966-68: Data, models, and evaluation of the arithmetic programs. New York: Academic Press, 1972.

Swanson, A. K. The computer as a tool of instruction: An annotated bibliography of selected 1972-73 periodicals. AEDS Journal, 1973, 7(2), 56-62.

Swanson, A. K. The computer as a tool of instruction: An annotated bibliography of selected 1972-73 books and reports. AEDS Journal, 1974, 7(3), 92-96.

Swanson, J. M., Ledlow, A., & Harris, S. Using omnitab interactively in a statistics laboratory. Paper presented at the second annual Conference on the Use of On-line Computers in Psychology, St. Louis, November, 1972. (ERIC Document Reproduction Service No. ED 116 606)

Swanson, J. M., Riederer, S., & Weekly, H. Using omnitab to teach applied statistics. Austin: University of Texas, May 1973. (ERIC Document Reproduction Service No. ED 116 608)

Tait, K., Hartley, J. R., & Anderson, R. C. Feedback procedures in computer-assisted arithmetic instruction. British Journal of Educational Psychology, 1973, 43(3), 161-171.

Tanis, E. A. A computer laboratory for mathematical probability and statistics. Proceedings of a conference on computers in the undergraduate curricula. Claremont, California: The Claremont Colleges, June 18-20, 1973. (ERIC Document Reproduction Service No. ED 079 985)

Tubb, G. W. Current use of computers in the teaching of statistics. Paper presented at the Computer Science and Statistics Annual Symposium, Gathersby, Maryland, April 1977. (ERIC Document Reproduction Service No. ED 141 109)

Wang, A. C. (Ed.) Index to computer based learning 1976 edition. Milwaukee: University of Wisconsin-Milwaukee, 1976.

Wassertheil, S. Computer assistance in statistics. Improving College and University Teaching, 1969, 17, 264-266.

Wegman, E. J. Computer graphics in undergraduate statistics. International Journal of Mathematical Education in Science and Technology, 1974, 5, 15-23.

Wegman, E. J., & Gere, B. H. Some thoughts on computers and introductory statistics. International Journal of Mathematical Education in Science and Technology, 1972, 3, 211-221.

Wikoff, R. L. Using the computer in basic statistics courses

Proceedings of a conference on computers in the undergraduate curricula. Iowa City: University of Iowa, 1970.

Williams, R. H. On the teaching of order statistics and supporting conversational computer programs at the precollege level. *Contemporary Education*, 1977, *48*(2), 66-72.

Zinn, K. L. Programming conversational use of computers for instruction. In R. C. Atkinson & H. A. Wilson (Eds.), *Computer-assisted instruction*. New York: Academic Press, 1969.