DOCUMENT RESUME

ED 292 252 EC 202 034

AUTHOR Frick, Theodore W.; And Others
TITLE Special Education Teacher Computer Literacy Training. Project STEEL. A Special Project To Develop and Implement a Computer-Based Special Teacher Education and Evaluation Laboratory. Volume II. Final Report.
INSTITUTION Indiana Univ., Bloomington. Center for Innovation in Teaching the Handicapped.
SPONS AGENCY Office of Special Education and Rehabilitative Services (ED), Washington, DC.
PUB DATE 30 Dec 86
GRANT G008301778
NOTE 246p.; For related documents, see EC 202 032-036.
PUB TYPE Reports - Descriptive (141)

EDRS PRICE MF01/PC10 Plus Postage.
DESCRIPTORS *Computer Literacy; *Computer Uses in Education; Curriculum Development; *Disabilities; Microcomputers; Programing; *Special Education Teachers; *Teacher Education; Word Processing
IDENTIFIERS *Project STEEL

ABSTRACT

The document is part of the final report on Project STEEL (Special Teacher Education and Evaluation Laboratory) intended to extend the utilization of technology in the training of preservice special education teachers. This volume focuses on the second of four project objectives, the development of a special education teacher computer literacy training package. The training program is a multi-media (lecture/print/transparency material) flexible curriculum in eight modules stressing current microcomputer technology and potential classroom applications. The major portion of the document consists of the training curriculum with modules on: introduction to microcomputers (history of computers and structure and operation); software applications (examples of uses in math, reading, spelling, and readability determination); introduction to word processing (educational and classroom uses of word processing); evaluation of educational software; introduction to authoring systems; BASIC programming; resource materials; and conclusions regarding microcomputer applications in special education. Also included are results of course evaluations during the three years of the course development and modification. (DB)

FINAL REPORT

Department of Education Grant No. G008301778

Project No. 029KH30094

VOLUME II

A SPECIAL PROJECT TO DEVELOP AND IMPLEMENT A COMPUTER-BASED SPECIAL TEACHER EDUCATION AND EVALUATION LABORATORY:

Special Education Teacher Computer Literacy Training

Principal Investigators:

Theodore W. Frick, Lewis J. Polsgrove,
and Herbert J. Rieth

Center for Innovation in Teaching the Handicapped

Dr. Lewis Polsgrove, Director and
Professor, Special Education

School of Education
Indiana University
Bloomington

OVERVIEW

This report describes developed products, research, and evaluation regarding the computer-based Special Teacher Education and Evaluation Laboratory (STEEL) at the Center for Innovation in Teaching the Handicapped (CITH), School of Education, Indiana University, Bloomington. Four major goals were achieved in Project STEEL:

I. Development, implementation, and evaluation of a microcomputer-based observation system for codification, storage, and summarization of special education trainees' classroom teaching performances (STEEL/MBOS);

II. Development, field testing, and evaluation of computer literacy training procedures and materials for preservice and inservice special education teachers (STEEL/COLT);

III. Development, implementation, and evaluation of a computer-based testing system for assessing teacher knowledge (STEEL/CBTS); and

IV. Development and preliminary evaluation of a computer-based information management system for storing and retrieving data on special education teachers' performances during their preservice training program (STEEL/IMS).

Comprehensive descriptions of each of these major accomplishments are provided in four separately bound reports (Volumes I through IV, respectively). A fifth separately bound report contains the executive summary of Volumes I through IV, and should be read first.

This document contains Volume II only.

Computer Literacy Training

Development of COLT modules. A major accomplishment of the STEEL project involved the development and formative evaluation over a three year period of a computer literacy training package (COLT) for special educatior teachers. The COLT program is a multi-media (lecture/print/transparency material) curriculum designed to teach fundamental computer literacy skills and to assess learner's acquisition of those skills through proficiency testing. The eight modules included in the COLT training package are designed to familiarize teachers with current microcomputer technology and potential classroom applications. They were intentionally designed to be flexible; thus, they can easily be adapted to the individual needs of participants and also easily modified to incorporate current developments in the fields of education and microcomputer technology.

In developing the content for the COLT modules, one of the initial tasks undertaken by the STEEL development team involved identification of preliminary instructional objectives for the computer literacy training component. After appropriate major objectives had been identified, the staff conducted a literature search related to the various topics included in the modules. The search revealed that there was little information concerning computer education of teachers and virtually none related to the needs of special educators in the literature at that time. Thus much of the material for the modules was drawn from the general literature, from past research conducted at CITH, and from tapping the human information resources available in the STEEL staff. After obtaining information from all of these sources, eight modules were compiled.

To formatively evaluate the COLT modules, special education courses were offered for credit on the Bloomington campus during three consecutive summer sessions. The first summer session course was entitled "Microcomputer Applications in Special Education," and covered the material included in the COLT modules, as well as utilizing several software programs developed at the Center for Innovation in Teaching the Handicapped. The course was revised based upon student evaluations obtained at the conclusion of the first summer session, and offered again the following summer. Based on teacher feedback following this session, CITH staff decided that it was important to present one of the modules (which had received consistently low ratings by students) in a more thorough manner. The third year of the STEEL project, CITH staff conducted an intensive course in an authoring (the Apple SuperPILOT) language. A total of 55 students were enrolled in the three courses.

All of the participants were graduate level students who taught during the school year. Students received over 56 hours of direct instruction (two 3 1/2 hour classes each week for eight weeks) in the form of lecture/demonstrations, discussions, and "hands-on" applied experiences. "Hands-on" experience was provided in 30+ hours of open-lab, tutorial sessions. Instruction was provided by members of the graduate faculty of the Department of Special Education with assistance from CITH's professional staff. Two major brands of microcomputers most likely to be found in public school classrooms, the Apple II and Radio Shack TRS-80, were available in sufficient numbers for use by the participants. Information was also provided regarding other major brands of computers to which the students might have access, and differences among the various models were discussed when appropriate.

Evaluation of the COLT modules used in the course consisted of: (a) an assessment of changes in student attitudes toward microcomputers as a result of participating in the course, (b) a determination of student mastery of the course content; and (c) a survey of students' satisfaction with the training experience. The results of these evaluation measures are presented in a separate section of this report.

Because of the increasing sophistication of classroom teachers regarding the use of microcomputers, the third course offered by the STEEL staff moved from the more comprehensive overview to a particular skill development format. In the previous two years, the BASIC programming module had consistently received the lowest evaluatory ratings. Project staff decided to present a more thorough presentation of a lesson authoring language that teachers could use to write computer programs which would supplement their regular curricular materials. The SuperPILOT language used on Apple microcomputers was selected because (a) formative evaluation information was available to use as a basis for improving content presentation, (b) many teachers had access to Apple microcomputers in their classroom and/or schools, and (c) the authoring language has had extensive revision and is a flexible yet failry easy to learn programming language. Each teacher attending the course developed an individually-designed software program and completed a criterion-referenced test demonstrating their mastery of SuperPILOT commands. Consumer satisfaction regarding the course is presented in the course evaluation section.

TABLE OF CONTENTS

MODULE 3: INTRODUCTION TO WORD PROCESSING

MODULE 4: EVALUATING EDUCATIONAL SOFTWARE

MODULE 5: INTRODUCTION TO AUTHORING SYSTEMS

MODULE 6: BASIC PROGRAMMING

MODULE 7: RESOURCE MATERIALS

MODULE 8: CONCLUSION

COURSE EVALUATIONS

ABSTRACT

COmputer Literacy Training (COLT) Modules

The COmputer Literacy Training (COLT) modules represent three years of development, field testing, and formative evaluation of the product, a computer literacy training package for special education teachers. The COLT is a multimedia curriculum designed to teach fundamental computer literacy skills through proficiency testing. There are eight modules, each dealing with a different aspect of microcomputer use. The modules were intentionally designed to be flexible; thus, they can easily be adapted to the individual needs of participants and also easily modified to incorporate current developments in the fields of education and microcomputer technology.

The modules are included in paper format, although in actual use transparencies, lectures, and "hands-on" experiences comprised a large part of the training strategy. Following the individual modules, course evaluations for each of the three years it was offered are included as one form of summative evaluation.

INTRODUCTION

Recent advances in computer technology have major implications for the career and job functions of the special education teacher. Microcomputers have been demonstrated to be effective tools enabling special educators to: provide handicapped students the opportunity for experience in computer use and programming; accurately assess and assist in remediating students' basic skills; develop Individual Education Plans (IEP's); monitor student academic performance; and provide feedback and reports to students, teachers, administrators, and parents. However, because of a lack of training, the majority of special education teachers are not able to take full advantage of the potential microcomputers offer.

In order to provide special education teachers with sufficient resources and experience in the use of microcomputer technology with their students, the Center for Innovation in Teaching the Handicapped (CITH) at Indiana University has developed the COmputer Literacy Training program. It is a hierarchical three-level microcomputer software and lecture/print material curriculum support package that will effectively teach fundamental computer literacy skills to teachers and assess those skills through proficiency testing. The package was developed as part of the federally-funded Special Teacher Education and Evaluation Laboratory (STEEL) grant. It consists of eight modules designed to familiarize teachers with current microcomputer technology and potential classroom applications. The primary focus is on the use of microcomputers for instructional purposes. The training sequence will train and evaluate special education teachers in the use of microcomputers for instruction of the handicapped as well as for

administrative purposes, e.g. IEP record keeping, determining readability of text material, testing students, etc. The modules are designed to be adaptable, responding to the individual needs of participants as well as current developments in the fields of education and technology.

The material in this manual was developed for implementation on three of the most popular microcomputers currently in common use within the public school systems: Apple II, TRS-80 and Commodore. "Hands-on", or interactive techniques for direct teacher contact with computer-related content materials are stressed. Training objectives have been classified into three levels to correspond to the three hierarchically ordered levels of training. Each level contains an integrated set of modules with related lecture/print materials which focus upon a sequenced approach to teaching computer literacy skills. Trainees move linearly through the levels of training based on mastery of preceding content determined through competency testing. The three levels of objectives are as follows:

1. Introductory Level Objectives

   a. To reduce initial teacher anxiety when first introduced to a microcomputer.
   b. To introduce special education teachers to the history of computers and of computers in education.
   c. To introduce special education teachers to general fundamental concepts of computer operation and functioning.
   d. To introduce special education teachers to the uses of microcomputers in education through programming.
   e. To assess teachers' skills in the above areas through computer-based competency testing.

2. Proficiency Level Objectives

   a. To teach special education teachers fundamentals of microcomputer structure and terminology.

b. To acquaint special education teachers with the instructional and managerial uses of the microcomputer.
c. To provide a guided "hands-on" BASIC language programming experience oriented toward educational applications.
d. To train special education teachers to trouble-shoot, or "debug", faulty BASIC programs.
e. To train special education teachers in the use of MENTOR, a portable, microcomputer-oriented, CAI applications development and administration system.
f. To assess special education teachers in these skills through computer-based competency testing.

3. Independence Level Objectives

a. To train special education teachers to operate microcomputers independent of continuous technical assistance.
b. To teach special education teachers how to use "canned" or packaged software programs.
c. To enable teachers to utilize a software development package to create educational application programs.
d. To provide information on external educational software resources.
e. To provide special education teachers with skills for evaluating microcomputer hardware resources.
f. To assess special education teachers in these skills through computer-based competency testing.

To use the COLT materials, the course instructor should review the Activity Sequence for each module, obtain and have available for use the necessary Instructional Materials, and become thoroughly familiar with the content. Additional information and/or materials may be required, depending upon the proficiency level of the students. After completion of the eight modules, student knowledge, degree of comfort in working with computer hardware and software, and computer expertise should show a significant increase.

SYLLABUS

Microcomputer Applications in Special Education

Instructor: (Name of instructor)

Purpose of Course

A considerable amount of attention is being given to the role of the microcomputer for educating children of all types. With school districts throughout the nation investing large amounts of money into purchasing hardware and software for classroom use, it will soon become mandatory for teachers to be skilled in the use of microcomputers. As in many areas of education, the microcomputer appears to have a bright future in the field of special education. Some possibilities or applications of microcomputer technology in special education include administrative data and record keeping, instruction, classroom-based data collection and analysis, and development of IEP's. This course is designed to prepare special education teachers to use and evaluate microcomputers and microcomputer software and to gain a familiarity with a variety of computer instructional programs.

Objectives

At the end of this course, participants will be able to:

1. Identify major milestones in the history of computers.
2. Define various computer-related terms.
3. Differentiate between various types of computers and their uses.
4. Identify various types of hardware and peripherals.
5. Connect and operate various types of computer hardware.
6. Provide a rationale for educational applications of computers.
7. Operate/use various educational software programs.
8. Evaluate various educational software packages using appropriate criteria.
9. Demonstrate how to perform major functions in operating a word processing program.
10. Use a word processing program to write a project report.
11. Demonstrate how to perform major functions in operating a lesson authoring system.
12. Use a lesson authoring system to produce a short software lesson for special education students.

13. Identify various types of commands in BASIC.

14. Copy and operate a simple program in BASIC.

Requirements and Activities

The following are examples of possible requirements and activities:

1. Classes will meet at the designated time: 1:00 p.m. to 4:30 p.m. on Tuesdays and Thursdays for instructional and laboratory sessions.

2. Out of class practice will be arranged on computers available at the School of Education, Main Library, and CITH.

3. Participants will be required to pass a criterion test on basic computer terms.

4. Participants will be required to review commercial software programs and turn in written evaluations of three of these.

5. Participants will submit a three to four page report prepared using a microcomputer word processor which identifies the instructional needs of children in their respective special education classrooms and the microcomputer hardware and software for use with these children.

6. Participants will demonstrate a simple CAI lesson which they have prepared with a lesson authoring system (SuperPILOT).

7. Participants will demonstrate a simple computer program which they have prepared from a written BASIC program.

8. Class will visit a commercial computer facility.

Evaluation Criteria for (Course Number) Enrollees

The following are examples of possible evaluation criteria:

1. Satisfactory completion of terminology criterion test	20 pts.
2. Evaluations of computer software programs	20 pts.
3. Word-processed report on hardware and software	20 pts.
4. CAI lesson prepared with SuperPILOT	20 pts.
5. Computer program written in BASIC	20 pts.
TOTAL	100 pts.

AGENDA

Microcomputer Applications in Special Education

MODULE	TOPIC
1	*Introduction to the program and distribution of syllabus. *Administration of assessment materials (attitude and terminology). *Discussion of the history of computers and the place of microcomputers. *Introduction to the general structure and operation of microcomputers. *"Hands-on" experience with software dealing with word-processing/computer graphics which have been designed as student motivational aids in the classroom (ComputerWRITER/DOODLE).
2	*Discussion of computer terms (focusing on the microcomputer). *Discussion of types of software, their applications and sources. *"Hands-on" experience with software dealing with remedial math and reading (CMMRS, CIRIS & SPELLMASTER). *"Hands-on" experience with software to assist teachers in determining the appropriateness of educational materials (CRIS).
3	*Introduction to word processing and its applications in the classroom. *"Hands-on" experience using and evaluating word-processing software for the APPLE.
4	*Discussion of criteria to use in evaluating commercial software designed for instructional use. *"Hands-on" experience using and evaluating commercial software designed for instructional use.
5	*Introduction to lesson authoring software. *Introduction to the APPLE SuperPILOT lessons which are oriented to a CAI application in reading or math.
6	*Introduction to the BASIC Programming Language. *"Hands-on" experience creating simple BASIC programs.
7	*Presentation by outside consultants (e.g. Indiana Clearinghouse for Computer Education) on services available to (your state) teachers. *Visit to a local commercial computer facility to gain familiarity with additional microcomputer brands and products.

MODULE	TOPIC
8	*Discussion of available microcomputer hardware and software resources suitable for education. *Discussion of packaged vs. teacher-developed educational software. *Discussion of issues involved and logistical considerations in the integration of computers in classroom environments. *Additional "hands-on" time with the SuperPILOT, Word Handler and packaged software. *Administration of assessment materials. *Closing remarks and workshop evaluation by teachers.

MICROCOMPUTERS IN SPECIAL EDUCATION

MODULE 1: Introduction to Microcomputers

While computers have been used in education for almost 30 years, the microcomputer is a relatively new technological advancement, one which offers great potential for instructional purposes. In order to fully appreciate the development and availability of microcomputers, it is desirable to trace the history of computers and explore the place of microcomputers in that history. Further, while many, or even most, microcomputer users will limit their involvement to running software programs developed by others, knowledge of the general structure and operation of microcomputers will aid in locating, evaluating, and operating appropriate software.

Objectives

At the end of this module, participants will be able to:

1. Identify the characteristics of the various generations of computers.
2. Distinguish between the structures and uses of mainframe, mini-, and microcomputers.
3. Identify the various hardware components of a microcomputer.
4. Identify the various types of microcomputer peripherals.
5. Identify the three major functions of microcomputer operations and identify the hardware and software components dedicated to each.
6. Distinguish between and give examples of low- and high-level programming languages.
7. Connect and operate various types of computer hardware.

THE PLACE OF MICROCOMPUTERS IN THE HISTORY OF COMPUTERS

A. INTRODUCTION

1. Mechanical devices designed to control a sequence of operations date back to the 17th century, when the rotating pegged cylinders still seen in music boxes today were developed. The first adding machines also appeared in the early 17th century, the most famous of which was invented by Pascal, a French scientist and philosopher.

2. It was not until the mid-nineteenth century, however, that adding machines were commercially available, on a limited basis. In 1834, Charles Babbage developed the concept of a program-controlled, mechanical, digital computer, incorporating a complete arithmetic unit, storage unit, punched-card input and output, and printing mechanism. Babbage called the machine an analytical engine and it was essentially programmed by rotating pegged cylinders that controlled the sequencing of other mechanisms.

3. For the next century other individuals worked alone and together on various adding machines, tabulating machines, and data sorters.

4. In 1943, IBM produced a large-scale electromechanical calculator called the MARK I. Still not strictly a "computer", the MARK I was a huge affair, capable of multiplying two 23 digit numbers in 6 seconds and was controlled by a sequence of instructions specified by a perforated paper tape.

B. FIRST GENERATION COMPUTERS

1. The first generation of computers began in the mid 1940's and lasted through the 50's. These computers are characterized by the use of vacuum tubes as the active components.

2. J. Presper Eckert and John Maunchly designed and produced the first totally electronic computer in 1946. It was called the ENIAC (Electronic Numerical Integrator and Calculator) and was intended primarily for army artillery calculations, even though it was really a general purpose device. The ENIAC had 19,000 vacuum tubes, was extremely expensive, and consumed an enormous amount of electricity. Even so, it could perform 5,000 arithmetic operations per second, approximately a thousand times faster than the MARK I. However, programming the ENIAC involved manual setting of hundreds of plugs and switches, which could take up to a day to do.

3. Even before the ENIAC was finished, the designers started to plan its successor, which they called the EDVAC. The EDVAC had a much larger internal memory than the ENIAC, yet was far more economical in its use of the vacuum tubes The EDVAC included only one-tenth of the equipment used in ENIAC, but provided 100 times the internal memory capacity and worked just as fast. Programming time was also decreased considerably.

4. Throughout the late 40's and 50's, many companies produced vacuum tube computers with further refinements on memory-storage systems and programming techniques. Some of these include the UNIVAC, produced by what is now the Remington-Rand Corporation; the EDSAC, from a group in England; the Pilot ACE, also from England; the SEAC, built under the direction of the National Bureau of Standards in Washington; the Whirlwind I, designed at M.I.T.; and others by such companies as IBM, Burroughs, NCR, and RCA.

5. An important development during the 1950s was to signal the end of this first generation of computers. The transistor was developed and the advantages of transistors over vacuum tubes for computer applications were almost immediately recognized.

C. SECOND GENERATION COMPUTERS

1. 1959 marks the beginning of the second computer generation. Transistors completely replaced vacuum tubes as the active computer components.

2. Many of the same companies mentioned earlier produced these new computers which were more economical, much faster, and much smaller than those of the first generation. IBM and CDC (Control Data Corporation) were two of the major suppliers of large computers during the 1960's, as they are today.

3. The models of transistor computers produced by various companies during this period are too numerous to mention here. It is sufficient to say that advances made during the second generation, convering roughly the period of 1959 to 1964, helped to firmly establish the utility and affordability of computers for large-scle data processing.

4. The first supercomputer was developed by Seymour Cray at CDC in 1964. While still a transistor computer, this CDC 6600 was EXTREMELY fast and was used for complex scientific work requiring very high precision and very high repetition calculation. These machines are still in use today and are still competitive, except that CDC has withdrawn much support in favor of more recent, even faster models.

D. THIRD GENERATION COMPUTERS

1. The advent of the integrated circuit in the middle 1960's marked the actual start of the 3rd generation which lasted roughly through the 1970's. An integrated circuit is an electronic device incorporating hundreds and now thousands of transistors and related devices in one small package. Each circuit performs a predefined function based on the larger circuit in which it will be placed. It consumes only miniscule amounts of energy, yet performs more functions per unit of space with greater reliability than previously possible.

2. A hugely successful computer produced at the start of this period is IBM's System 360, followed in 1970 by their 370 line, both of which are still in use in various parts of the country. Many features from these lines have become standard in the computer industry. New memory storage techniques and much faster capabilities, along with reduced costs, are characteristic of third generation computers.

3. A plethora of general-purpose and special-purpose computers were introduced during this period. Major companies included CDC, Sperry-Rand, Burroughs, Honeywell, General Electric, and DEC (Digital Equipment Corporation).

4. DEC also played an important role in the growth of the mini-computer field in the latter part of the third generation. Their PDP series is representative of the medium-scale computers which utilize large-scale integration technology in physically small spaces. A mini-computer, the size of a storage cabinet, is able to perform the functions of earlier, much larger, computers, yet at a much faster speed.

5. Another important development during the latter part of the third generation was the introduction of the microcomputer. This was made possible by the development of the microprocessor integrated circuit chip in the late 60's. The first chips were somewhat crude, but hobbyists designed their own very simple microcomputers from them. The microcomputers were very expensive and had no readily available disk storage systems.

6. In the early 70's, companies began marketing microcomputers to persons with specialized knowledge. The INTEL 8080 microprocessor was a popular chip of the day.

7. The Radio Shack TRS-80 Computer Model I was the first consumer marketed microcomputer using the Zilog Z80 microprocessor chip. Other companies currently marketing microcomputers suitable for use in education include: APPLE, ATARI, Ohio Scientific, Heath Company, COMMODORE (PET, VIC, and CBM Machines), and IBM, among others.

8. The microcomputer represents a giant step in the computer field. What used to be accomplished in the first generation and some second generation computers, taking up whole rooms, can now be accomplished in a desk-top computer which operates many times faster and at an EXTREMELY lower cost. In addition, increased memory storage capacities allow for vastly increased data-processing capabilities.

E. FOURTH GENERATION COMPUTERS

1. Computers are now in their fourth generation. There is no one development which can be said to end the third generation and begin the fourth. Basically, fourth generation computers are characterized by very large-scale integration. This involves grouping an assortment of functions on a single chip, as opposed to the single function chips used in third generation computers. Large-scale integration allows further advances in storage capacities, programming techniques, and operating speeds.

2. Large-scale computers, mini-computers, and microcomputers all have their place in meeting today's computing needs. Large-scale computers, such as the CDC Cyber 170/855 and the DEC System 2060, are used for scientific applications such as complex statistical analyses or management of large databases or information files. Also called mainframes, these computers can serve hundreds of users simultaneously and are primarily found in large institutions, such as corporations, universities, and governmental agencies.

3. Mini-computers, also called mid-size, are capable of the same operations as the mainframes, but at a slower speed, and are used in situations where the computing load is lighter. Mini's, such as the DEC VAX 11/780's and the IBM 4341, are used for such purposes as management of smaller databases, electronic mail, word processing, and as assistant computers to distribute the workload across larger machines. Mini's areappropriate for mid-size businesses and can serve from 16 to 64 users simultaneously.

4. Microcomputers have primarily been designed for the single user, but later models can accommodate up to 12 users at one time. Micros have multitudes of uses:
   --In the home they are used to control appliances and to manage budget and tax information, as communication terminals to information systems, and for entertainment.

   --Small businesses utilize them for word processing, inventory control, personnel and financial record-keeping, business forecasts, and order processing.

--One of the most exciting uses of the microcomputer, however, is in education, not as a replacement for the teacher, but as an efficient and motivating instructional aid. In the classroom, the micro can be used for remediation of basic skills, curriculum enrichment, routine drill and practice, reinforcement for desirable performance or behavior, student record-keeping, IEP development and management, and preparation of instructional reports. It is the use of the microcomputer in the classroom that will be the focus of this course.

F. SUMMARY

1. Technology has made great strides since the development of the first vacuum tube computers in the early 40's. It is not unreasonable to predict that many refinements and "breakthroughs" are still on the way. The next decade will probably produce a further decrease in the size of computer components, along with an increase in the efficiency and capabilities of the machines. New programs will be developed to broaden the range of applications and expand the usability of the technology. Costs will continue to fall and more and more "ordinary" individuals will find themselves affected by and utilizing this technology.

2. Designers are already looking ahead to the fifth generation computers which are likely to utilize voice input and voice output to control operations. This advancement will no doubt greatly enhance the usability of computer resources by a larger proportion of our society.

Figure 5. ELECTRONIC COMPUTER GENERATIONS

DEVELOPMENT	1st GEN. 1946-1953	2nd GEN. 1959-1964	3rd GEN. 1964-Late '70s	4th GEN. Late '70s---
COMPONENT TECHNOLOGY				
Vacuum Tubes	(----------)			
Transistors		(----------)		
Integrated Circuits			(----------)	
MAIN MEMORY TECHNOLOGY				
Delay Lines	(----------)			
Electrostatic Tubes	(----------)			
Magnetic Drums	(----------)			
Magnetic Cores		(----------)		
Solid-state Large-scale Integration			(----------)	
MAIN MEMORY CYCLE TIME				
40 - 40,000 μs	(------)			
10 - 20 μs		(-----)		
2 - 10 μs			(----------)	
0.5 - 2 μs			(------)	
0.02 - 1 μs				(-----)
0.005 - 0.02 μs				(----------
PERIPHERAL STORAGE				
Magnetic Tapes	(----------			
Magnetic Drums	(----------			
Magnetic Disks		(----------)		
Laser and Magnetic Bubbles			(----------	
TYPICAL EXAMPLES:	ENIAC, EDVAC UNIVAC, I,II Pilot ACE SEAC, EDSAC Whirlwind I IBM 700's, 650 Datamatic 1000	Philco 2000 CDC 1604, 3600 IBM 7000, 1400 Ferranti Atlas RCA 301,501 Honeywell 800 UNIVAC III	CDC 6000 IBM 360,370 UNIVAC 1108 1110 CDC Cyber 70 DEC PDP-10 Honeywell 200,2000	CDC 170, 205 IBM 3030 CRAY 1 DEC PDP-20, VAX 11 Microcomputers

Adapted from Encyclopedia of Computer Science, First Edition. A. Ralston and C. L. Meek (Eds.). New York: Van Nostrand Reinhold Company, 1976, 474-494.

STRUCTURE AND OPERATION OF MICROCOMPUTERS

"Everything a computer does, it does from a program. Every program is initially written by a human being. Therefore, everything a computer does, right or wrong, it does from a program written by a human being." (provided it is working as designed) - Robert Eckert

"If these kids can do it, I can do it." - anonymous teacher

A. INTRODUCTION

1. Microcomputers, like all computers, are simply tools which provide a means to an end. They accept, manipulate, report, and store information given them by the user. These functions are accomplished by the machine following a set of instructions called a program. These programs are ultimately resolved down to a series of numbers which stand for each of the instructions the machine is to perform. Most microcomputers use BASIC as the primary language for their programs. Some also support other languages such as COBOL, Pascal, FORTRAN, and LOGO.

2. Most microcomputers fit on a desk-top, although some are now small enough to use on a person's lap. Despite their small size, these machines can accomplish more tasks, faster and at an extremely lower cost, than most first and second generation computers which needed large rooms to house all their components.

3. Costs of microcomputers range from $50 to $20,000, depending upon the size of internal memory and the speed of the system. Home computers generally run from $50 to several thousand dollars. Business microcomputers usually cost from $1,000 to $20,000.

B. STRUCTURE

1. The components of a microcomputer are shown in Figure 6. They consist of a KEYBOARD or other device for communication or input, a MONITOR or television for visual display, DISK DRIVES or other devices for external memory storage, INTERNAL MEMORY storage and a CENTRAL PROCESSING UNIT, and, optionally, a PRINTER for production of hard copies of the output. These physical components are called computer HARDWARE. They are the units which execute the instructions provided by computer programs. These sets of instructions or programs are called SOFTWARE. Each of the hardware components are described below.

2. The major component of a microcomputer is the central processing unit (CPU). The CPU generally consists of a single microprocessor chip (see Figure 7) or a number of separate chips. A microprocessor chip is a wafer slice, usually of silicon in combination with overlays of semi-conducting materials, containing an integrated circuit or integration of many circuits. These circuits contain the essential elements of the central processor, including the control logic, instruction decoding, and arithmetic-processing functions. The CPU can be thought of as the "brain" of the computer and is always located within the machine itself.

3. INTERNAL MEMORY also consists of microchip(s), located within the machine, which are used by the CPU to hold your program and its data while it is running it, and to contain the computer's basic operating functions. Memory can be thought of as little 'mailboxes', each with an address and a contents on a long street, into and from which the CPU places and retrieves information.

4. The KEYBOARD, which operates much like that of a typewriter, is the user's means of interacting with the computer's CPU. The keyboard allows the user to enter data and program commands which instruct the CPU what functions to perform. Joysticks, voice input and other devices such as graphics tablets, light pens, and touch screens perform the same function as a keyboard.

5. The MONITOR or television provides a means of visually displaying entry data, intermediate program steps as they are executed, and program results.

6. The DISK STORAGE UNITS are used for external storage of the software or programs which contain the instructions for the CPU. The programs are stored on diskettes which are manually inserted into the disk drive units. The most commonly used diskette is a 5 1/4" floppy disk made of a pliable substrate covered with a material which can take a magnetic charge. Larger and smaller floppy and hard disks are also used by some disk units. Not all microcomputers use disks for data and program storage. Some use ordinary audiotape cassettes in a special cassette recorder; others use simple plug-in cartridges.

7. A PRINTER is a useful accessory to any basic computer set-up. When directed by the CPU, the printer produces a hard copy of computer output results for visual examination or storage. There are four basic types of computer printers: dot matrix, daisy wheel, thermal and ink jet. With a dot matrix printer, a set of wires in the print head strikes the printer ribbon to form a character. The character is thus composed of tiny dots separated by spaces. While dot matrix printers are less expensive, they generally do not produce as sharp a copy as

daisy wheel printers. Daisy wheel printers use a solid type device to press the ribbon against the paper, producing a printed copy that looks just like a typewriter original. These printers are more expensive than dot matrix printers, but produce a more crisp, clean copy. Thermal printers, like the dot matrix, use a set of wires in the print head. However, instead of striking a ribbon, the wires generate heat which produces a dot image on specially prepared paper. Finally, ink jet printers spray ink onto high fiber paper to produce solid characters which appear much like those produced by a typewriter. Although all types of printers are constantly being improved, dot matrix and thermal printers generally produce what is called "draft quality" results, while daisy wheel and ink jet printers generally produce higher "letter-quality" results.

8. The CPU and the internal memory comprise the computer itself. All the other devices described above are actually PERIPHERALS, or machines that operate in combination or conjunction with the computer. Some brands of microcomputers package each of these components separately; others will provide all or most devices in one structure. The printer is considered an option and is usually sold separately.

C. OPERATIONAL FUNCTIONS

1. Microcomputer operations can be characterized as involving three functions: input, process, and output (see Figure 8). INPUT is the process of entering information into the computer for subsequent action by the CPU. Input can come from programs stored on external memory devices such as a disk drive, cassette recorder, or program cartridge. Input can also come from such immediate communication devices as the keyboard, joysticks, light pens, a "mouse", or a voice input device.

2. The intermediate function between input and output is the processing function. Information fed into the microcomputer is processed through the CPU. The CPU does only what it is told to do by a stored program or by the user through an immediate communication device. The CPU can perform any number of functions including: using a special program to translate another program into a language form that the computer can manipulate (machine language); perform arithmetic calculations; place input or output information in the appropriate internal or external memory locations; and signal the appropriate connected peripherals to perform their specialized functions. Internal memory is an essential component involved in the processing function. This memory holds the instructions and data upon which the CPU acts. The different kinds of internal memory (ROM, RAM, PROM and EPROM) and the means by which information is represented in memory will be discussed in Module 2 of this series.

Figure 8. Microcomputer operations

INPUT ⟶	PROCESS ⟶	OUTPUT
EXTERNAL MEMORY: Disk Drive Tape Recorder Cartridge IMMEDIATE COMMUNICATION: Keyboard Joysticks, etc. Voice Control	CENTRAL PROCESSING UNIT (CPU) INTERNAL MEMORY: ROM RAM PROM EPROM	DISPLAY MONITOR TELEVISION PRINTER VOICE SYNTHESIZER CRT TERMINAL

From Florence M. Taber (1983). Microcomputers in special education: Selection and decision making process. Reston, VA: The Council for Exceptional Children, p. 6.

3. OUTPUT is the results of the actions taken by the computer's CPU. Output is usually viewed on the monitor or television set connected to the computer, but can also be printed out as hard copy from a printer, or spoken orally through a speech synthesizer. Output can also be transmitted to a remote location such as a printer or CRT TERMINAL (an interactive station consisting of a keyboard and monitor connected by phone wires or coaxial cables to a remote computer.) That remote location can also be another microcomputer programmed to receive, display, manipulate, and/or store the transmitted output.

D. PROGRAMMING LANGUAGES

1. Computers operate by following instructions which have been stored inside previously by humans or which are given by the immediate user. Since most computers cannot yet interpret human speech, these instructions must be written in a language which is understandable by computers. These languages can be divided into three major categories: MACHINE LANGUAGE, ASSEMBLY LANGUAGE, and HIGH-LEVEL LANGUAGES.

2. MACHINE LANGUAGE uses numbers as codes for the desired operations. These codes are directly interpreted by the electronic circuitry of the computer and cause the operations to occur. Machine language essentially tells the computer which internal switches to turn on and off to represent data and instructions. Programmers never use this language to write computer programs because it is extremely tedious and time-consuming to do so.

   Example: The English instruction "assign the value of one to a variable called 'output'" could be represented in machine language by:

```
0500 00 00 0 000000 0000
1000 00 00 0 000004 0000
0100 00 00 0 000001 0000
```

3. ASSEMBLY LANGUAGE is a language in which mneumonics or alphabetic codes are used to represent the number codes in machine language. While it is easier to program in assembly than in machine language, assembly is not generally used in applications programming unless some portion of a program requires maximum speed by the CPU.

   Example: The English instruction "assign the value of one to a variable called 'output'" could be represented in assembly language by:

```
SZ   TEST
LA   A0,(1.000000+ 0)
SA   A0,OUTPUT
```

4. HIGH-LEVEL LANGUAGE is a term for those programming languages that resemble human language and cannot be interpreted directly by the computer's electronic circuitry. We are able

to program in high-level languages because programmers have developed assembler, compiler and interpreter programs which translate the high-level language ultimately into the machine language that the computer can interpret and act upon. These programs are either stored in the microcomputer internal memory or on the software diskettes the programmer uses to develop a program in the high-level language. There are many different high-level languages that have been developed for various applications. While most microcomputers support the programming language BASIC, others also support FORTRAN, Pascal, COBOL, and others.

Example: The English instruction "assign the value of one to a variable called 'output'" could be represented in the high-level language BASIC by: OUTPUT=1

GLOSSARY OF COMPUTER TERMINOLOGY

ASSEMBLERS: Since writing a program in the most primitive binary instructions that the computer can process (see MACHINE LANGUAGE) would be extremely tedious, programmers developed "assembler" programs which convert mneumonics for the instructions into the actual binary instructions. Using mneumonics speeds program development and error correction.

BACK-UP: A duplicate copy of a file or program. You make back-ups of material on disk or cassette in case something happens to the original.

BASIC: Beginner's All-purpose Symbolic Instruction Code. A high-level language used in most personal computers and fairly easy for programming novices to learn.

BATCH MODE: A term used to describe the use of a computer in which users supply multiple tasks for the computer to accomplish in a "batch" which are then processed by the machine and the results or output is later forwarded to the user.

BAUD RATE: Rate of transmission or reception of information by a particular machine.

BINARY: A numbering system based on 2's rather than 10's, which uses only the digits 0 and 1 when written.

BITS AND BYTES: Bits are the smallest memory units of a computer. A single bit can be either "1" or "0". Bytes are the smallest meaningful memory units usually composed of 8 bits in smaller machines. The number of bytes available for storage is usually expressed in thousands,or "kilobytes"; Thus "16K" refers to 16,000 bytes of available memory, "32K" to 32,000 bytes and so on.

BOOT: To ready a computer for use by loading the disk operating system into the computer's temporary memory, or RAM.

CATHODE RAY TUBE ("CRT"): Remote computer terminal consisting of a keyboard and visual monitor that allows a user to interact with the computer.

CENTRAL PROCESSING UNIT (CPU): The central processor of the computer system. It contains the main storage, arithmetic unit, and special register groups.

CHARACTER: One symbol of a set of elementary sumbols, usually including the decimal digits 0 through 9, the letters A through Z, punctuation marks, operation symbols, and any other single symbols which a computer may read, store, or write.

COMMAND: An order to the computer to execute a task.

COMPATIBILITY: This term refers to hardware components, software, or peripheral equipment devices which can be used on more than one system with little or no adjustment.

COMPILERS: Advanced assembler programs which convert English-like languages such as BASIC into the binary instructions used by the computer. The operator's program is compiled as one entire unit, the resulting instructions are then executed.

COMPUTER CRASH: A loss of information due to a machine failure or loss of power during data storage or manipulation.

COMPUTER-ASSISTED INSTRUCTION (CAI): Instruction of students that is largely under the control of a machine including presentation and pacing of the content branching, remedial loops, drill, practice, testing, monitoring, and record keeping functions.

COMPUTER-MANAGED INSTRUCTION (CMI): A term usually referring to diagnostic assessment and placement of a student within a particular instructional program as well as continual testing as a method of evaluating student progress. The teacher controls both the content and the rate of learning.

CONTROL-KEY: Key that executes commands, in conjunction with other keys pressed simultaneously.

COPY: To duplicate a file or program so that you can retain the original and work on the duplicate. Usually refers to duplicating one disk to another.

CURSOR: A patch of light or other visual indicator on a screen that shows you where you are in the text.

DATA BASE: A file of information, abstracts, or references on a particular subject or subjects. Computer data bases may be continuously updated and are designed so that by using key words or phrases, users can quickly search for, sort, analyze, and print out selected data.

DATA BASE MANAGEMENT SYSTEM (DBMS): A software product which provides a systematic approach to storing, updating, and retrieval of information stored in a data base.

DATA PROCESSING: A term which refers to the handling of raw data according to precise procedures to accomplish such operations as classifying, sorting, calculating, summarizing, and recording.

DIGITAL COMPUTERS: A term pertaining to the functioning of the circuitry of virtually all computers from large "main frame" machines to microcomputers. Information storage and manipulation corresponds to "on" or "off" (1 or 0) switching within the circuitry.

DISK DRIVE: The slot or peripheral box(es) in which an operator places a diskette for operation of the machine or for storage of information.

DISK OPERATING SYSTEM (DOS): The program that instructs the computer's CPU how to transfer information to and from a disk. Each brand of microcomputer generally uses its own version of an operating system (e.g., IBM uses PCDOS, Radio Shack uses TRSDOS, APPLE uses APDOS.) Thus software written for one machine cannot generally be used on another brand. However, Digital Research Corporation has developed an operating system called CP/M (Control Program for

Microcomputers) which enables many different brands of microcomputers to share software.

DOCUMENT DISK: A term describing a diskette on which data is stored. It usually does not contain programs and may be erased for reuse.

DOUBLE DENSITY: A way of putting information on a disk that allows the disk to store twice as much data as a single-density disk.

DOWNLOAD: Loading information from a distant source to your particular terminal or computer.

FILE: Any data designated to stand alone. A file can consist of a program, a single name, an equation, a form letter--even an entire book.

FLOPPY DISK: A small, pliable disk (about the size of a 45 rpm record) with an enlarged center hole on which programs or data information is magnetically recorded for use by and with the microcomputer.

FRIENDLINESS: How easy a program or computer is to work with. A 'user friendly' program is one that takes little time to learn, or that offers on-screen prompts, or that protects the user from making disastrous mistakes.

GLITCH: An electrical pulse or burst of noise that causes errors, types of computer crashes, or failures. A glitch is beyond the user's control.

HARD DISK: An inflexible, metallic disk primarily used for auxillary storage of data. Hard disks provide much more storage area and faster access to data when compared to floppy disks.

HARDWARE: The actual machinery or electronic components responsible for the functions of the microcomputer, e.g.: keyboard, the video display screen, the disk drives, or any other additional components (peripherals) such as tape recorders, printers, graphics board, etc.

INITIALIZE: To reset the computer and its peripherals to a starting state before beginning a task. Done automatically by the disk operating system.

INPUT: Input refers to information or data transferred from an external storage medium into the internal storage of the computer.

INTERFACE: A device or program that permits one part of a computer system to work with another, as when making a connection between a cassette tape recorder and the computer.

INTEGRATED CIRCUIT CHIPS: The electronic components made primarily of silicon on which the memory and micro-processor circuitry are etched. Memory chips can be added to some machines to increase their memory capacity.

INTERACTIVE MODE: A term refering to use of the computer to interact ("talk") with the machine, usually from a remote terminal using pre-defined commands, to obtain instant feedback, information, and data analysis services.

INTERPRETERS: Programs which interpret the operator's program one command or statement at a time during its execution, compiling each statement into the binary instructions and then executing those instructions.

K: One kilobyte, or about 1,000 bytes, of memory.

LOAD: To enter a program into the computer from cartridge, cassette, or disk.

MACHINE LANGUAGE: Each computer processor whether large or small has a set of primitive instructions which it can execute to manipulate data. This is the computer's machine language. Each instruction has a unique binary number. Programs are constructed by sequencing the correct instructions in the correct order. Programming in machine language is tedious so other languages more easily used by humans were written in machine language. These languages are called "high level" or "user languages." Some common examples of user languages include FORTRAN, BASIC, LOGO, PASCAL, COBOL, AND PILOT. Each of these languages has an ideal use for a particular type of programming need.

MAIN FRAME COMPUTER: The fundamental portion of the computer that controls the CPU and control elements of a computer system. This term is usually used in reference to the large capacity computers in contrast to 'minis' and 'micros.'

MEMORY: The part of a computer that stores information for quick retrieval by the CPU.

MENU: A video display of tasks the computer can be ordered to perform. You choose the task by moving the cursor or entering an appropriate command.

MICROCOMPUTER: A small, portable and relatively inexpensive computer which incorporates many basic features of larger multi-user computers, except in a single-user package. It has capabilities for data and text processing, as well as game and educational software. Permanent storage of programs and data is usually accomplished with floppy disks or cassette tapes.

MINI COMPUTER: A mid-size computer characterized by higher performance than microcomputers, richer instruction sets, higher price, and a proliferation of high-level languages, operating systems, and networking methodologies.

MODEM: A device that allows a telephone hookup of a smaller computer or remote terminal to another computer. Information is transmitted through telephone lines between the machines.

OFF-LINE: A term used to describe a system or peripheral device in a system in which the operation of the peripheral equipment is not under the control of the central processing unit (CPU).

ON-LINE: A term used to describe a system or peripheral device in a system in which the operation of the peripheral equipment is under the control of the central processing unit (CPU). Information reflecting current activity is introduced into the data processing system as soon as it occurs.

OPERATING SYSTEM: The computer's internal program that oversees the movement of information between the CPU, the input/output devices, and a storage device such as cassettes or disks.

OUTPUT: Computer results, such as answers to mathematical problems, statistical or accounting figures, production schedules, etc. Output also refers to information transferred from the internal storage of a computer to secondary storage.

PERIPHERAL: A term pertaining to any equipment external to the computer such as a keyboard, printer, plotter, touch screen, graphics board, or cassette

PORT: The gateway that connects the computer to its outside world.

PRINTERS: A peripheral device that, when directed by the computer, produces a printed copy of program results. There are basically three types of printers that will produce hard copy from a source machine: thermal, dot matrix, and character.

PROGRAM: Any set of instructions that tells a computer what to do.

RANDOM ACCESS MEMORY (RAM): Memory used for the temporary storage of programs or data, etc. created or used by the operator while the computer is in use. Material created in RAM may be permanently stored on a diskette.

READ ONLY MEMORY (ROM): The computer's permanent built-in memory which contains the basic instructions that enable programs to be written and executed.

RESET: See initialize.

SAVE: A command to the computer to store completed work on tape or disk.

SOFTWARE: The programs that control the operations of the machine. Some examples include wordprocessor, data analysis, operating system, chess, "space invaders", or educational programs.

SYSTEM DISK: A disk on which programs are usually stored, the program "software". This disk also usually contains a special program written by the manufacturer of the computer which enables the computer to write and read information to and from external devices such as disk drives, printers, modems etc.

TIME SHARING SYSTEM: An arrangement whereby the computer processor is programmed to enable many users to use it at the same time with each person at a separate terminal.

UPLOAD: Sending information from your terminal or computer to another source, usually a large central computer.

WINCHESTER DRIVE: A form of hard disk permanently sealed into a case.

WORD PROCESSING: A subset of data processing operations such as text entry and editing, information management systems, translation, and typesetting. Word processing applications include the preparation of general correspondence, form letters, manuscripts, and standard reports.

TEN COMMANDMENTS OF FLOPPY DISK CARE

by

Anne Beversdorf

The one part of your computer system that is most vulnerable to damage, and is the most frequent cause of system breakdown, is the floppy diskette. Human hands can do no damage to the keyboard or to the monitor (or screen), but can wreak irrecoverable damage to the floppy diskette.

To protect your computer hardware you need do no more than eliminate foods or liquids from the computer area, turn off the machine before plugging in or unplugging any device, and keep the computer at temperatures in your comfort range and in a clean environment, so dust and dirt don't gum up the works. To be really secure, invest in a cover for the computer, and in some kind of circuit-breaker surge protector to protect the computer from bolts of lightning that could fry its insides. The disk drive itself is a slightly different story. The disk heads are very sensitive--just as are the heads of your expensive audio or video tape recorder--and can be damaged by dirty or damaged disks, or by inserting or removing diskettes when the heads are reading or writing information to the disks. To be safe, never insert or remove a diskette when the disk drive is "whirring," or when the warning light is on. To do so is something like scraping a needle across a record on your stereo turntable. It will damage both the diskette and the disk drive head.

That takes care of the hardware. But to protect your system from breakdown, you must apply constant vigilance to the care of your floppy diskettes. These little pieces of plastic and mylar contain information critical to making your computer perform. Diskettes are made of exactly the same material as audio tapes, but in a different shape. In both cases, the mylar base is coated with oxides that record information magnetically. Just as with an audio tape, small scratches or dirty spots on the surface can damage the information on the tape--but with an important difference. When you have a tape recording of your favorite album and someone's greasy finger has pinched the tape surface, you may hear a funny little "glitch" as you play the tape, but the tape will keep on playing. Pretty soon the next song will start and you'll learn to ignore the bad spot on your recording.

Unfortunately, computers aren't as forgiving as audio tape players. Your computer diskette contains thousands of tiny steps of instructions the computer must follow exactly in order to accomplish anything. Imagine trying to explain to some really stupid Alien Being every step you take in getting to school on Monday mornings: First you hear the alarm . . . then you reach to the left until you find the square box that is your alarm clock. Then you find the small button to stop the sound and give you ten extra minutes. You push the button, remove your hand and return to sleep. The alarm rings again. You go through the same process, realize it's the second time so you don't get ten minutes, so you roll over, push off the covers, bring your right foot over the side of the bed, then the left foot. You stand up, turn to the right, reach forward to the right for the covers, lift them, place

them over the pillow. Oh---forgot to mention straightening the pillows--that's lift pillow, place it to left side, match right angle of pillow to right angle of bed corner . . . Then you turn left, walk twelve steps, turn right, walk fifteen steps, and turn left (the bathroom) . . . you get the idea. If you leave out one tiny step, the Alien (read The Computer) has no idea what to do next.

This can all be translated into the way the computer reads information from the diskette. One single little scratch, greasy fingerprint, or dirt spot is likely to wipe out one tiny step of the computer's instructions. Without that single, tiny step, the computer will simply come to an absolute halt, totally ignorant of what to do next. Obviously, a totally ignorant computer trying to read information from a damaged diskette adds up to a lot of trouble for you, the computer user.

The best cure for computer diskette problems, to mangle an old metaphor, is a pound of prevention. So here is a list of what to do and what not to do when handling computer diskettes. Remember, even though the blank diskette may cost only $2.00, the information on that diskette may have cost hundreds of dollars, or hours and hours of your time. It's definitely worth learning the correct handling procedures.

Commandment One

NEVER touch the exposed surface of the diskette. The oblong hole in the plastic cover is where the disk drive's read-write head looks for and records information. This "window" to the diskette is where dirt or fingerprints are carried into the disk jacket where they can explode to monstrous proportions. If you know a diskette has been subjected to potential damage, make an immediate back-up copy. The damage may not show up immediately, but the dirt that was introduced will scrape off on the inside of the disk jacket and, the next time you pick the diskette up in exactly that spot, may get ground into the oxide surface. This can scrape off the exact portion of "instructions" that keeps your program running.

Note: if you have a single-sided diskettes, the disk drive is reading the unlabelled side. This seems backwards, so I'll repeat. The disk drive reads the unmarked side of the diskette. When your drive doors open horizontally or vertically, the open door will be on the labelled side. The drive reads the other side. WHEN IN DOUBT, PROTECT BOTH SIDES!

Commandment Two

PROTECT your diskettes from dust, dirt, smoke, paper particles (as found near printers or in school duplicating rooms), chalk dust, rubber eraser particles, food crumbs, liquids, peanut butter and jelly, pet-hair, and every other possible source of disk pollution.

Commandment Three

ALWAYS keep disks in their paper jackets, except when they are actually in your disk drive. This is the best way to protect them from dirt and scratches.

Commandment Four

NEVER use paper clips, rubber bands, staples, or clip-boards to attach diskettes together or to other documents. The pressure or crimping can chip the oxide surface and/or grind in dust particles that will chip or damage the surface.

Commandment Five

PROTECT diskettes from crimping or bending. This means transporting them in protective cases (the special plastic cases that hold ten disks at a time are the best protection for loose diskettes). NEVER carry a loose diskette in a purse, pocket, or knapsack. You are inviting trouble as certainly as did Lucy when she welcomed Dracula. When desperate, you can carry a single diskette, loosely, inside clean pages of a thin, hardbound book (a child's picture book is ideal--a heavy book might grind in dirt particles).

Commandment Six

CAREFULLY insert diskette in the disk drive door. Slide them in gently, the label side up, with the oblong window entering the drive first. If it doesn't go in immediately, make sure the drive is empty of another disk, then try again, slowly and gingerly. NEVER shove or force a disk into the drive. You will bend it and damage information. NEVER close the disk drive door on a carelessly inserted diskette. This can cause the clamping device to smash or bend part of the diskette rather than closing down over the disk hole. A damaged "hub" (the part of the disk around the hole) will make it impossible for the disk drive to read or write information accurately.

Commandment Seven

NEVER use a pencil or ballpoint pen to label a diskette. The pressure of a pencil or ballpoint can damage the oxide surface. ALWAYS use a felt-tipped pen for labelling. NEVER use a pencil or ballpoint on paper resting on a diskette. The pressure can carry through the paper and the disk jacket to cause damage.

Commandment Eight

THIS SECTION IS CRITICAL!!! AVOID MAGNETIC FIELDS!!! Keep diskettes at <u>least</u> two feet from magnets--key chains, paperclip holders, toys . . . AVOID TELEPHONES! Phones have powerful electromagnetic coils that come alive when the phone rings. AVOID STEREO SPEAKERS! Stereo speakers, radio speakers, and other audio devices are sources of magnetism. So are many other "innocent" electrical appliances. Keep your diskettes away from them. NEVER place diskettes on top of your monitor, screen, or TV. Video sources contain electromagnetic waves and can change or destroy information on your diskettes. There is more magnetic "shielding" on the thickest side of the monitor case, usually the side with the control knobs. Whenever possible, place your diskettes and disk drives on the "thick" side of the monitor.

Commandment Nine

STORE diskettes vertically, away from direct sunlight, heat sources, or excessive cold. Disks are comfortable in temperature and humidity ranges humans also prefer. If diskettes are transported in very cold weather, allow them to come to room temperature before using them.

Commandment Ten

NEVER try to use a visibly dirty disk. You will damage your disk drive heads and every disk you use thereafter. If you suspect disk damage, immediately salvage as much information as possible by transferring it to a new diskette. DO NOT try to clean disks with alcohol, thinners or freon. These solvents can dissolve the bond holding the oxide to the mylar surface, destroying your disk. In dire emergencies (an irreplaceable disk is covered with Coke syrup, for example) you can rinse gently with lukewarm water, twirling the disk inside its jacket to expose the dirty spots. Allow the diskette to air-dry completely, including the padding inside the disk cover, before using it. This will sometimes work, at least temporarily.

Now that you know the Ten Commandments of Diskette Care, you should be able to avoid many of the traumas of new computer users. These rules will help eliminate one of the factors in a surprise system failure. Make sure your students, colleagues, and friends who handle your diskettes follow these same rules and your diskettes should give you long, trouble-free service.

* Anne Beversdorf, Indiana Clearinghouse for Computer Education, IUPUI, Indianapolis, Indiana.

NAME_____

MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION

1. True or False:

____ One byte of memory is a unit of 8 bits in many microcomputers

____ One bit of memory is a unit of 8 bytes in many microcomputers

____ 16K refers to 16,000 bytes of memory

____ 32K refers to 32,000 Kilobits of memory

____ In a microcomputer, information is stored in silicon-based microchips.

2. Memory that is allocated for the temporary storage of programs and data is called: (circle letter)

a. Read Only Memory (ROM)
b. Erasable Memory (EM)
c. Programmable Read Only Memory (PROM)
d. Random Access Memory (RAM)
e. Random Acceptible Memory (RAM)

3. The nonprogrammable memory containing the machine-based instructions is called: (circle letter)

a. Read Only Memory (ROM)
b. Erasable Memory (ER)
c. Programmable Only Memory (POM)
d. Random Access Memory (RAM)
e. Random Acceptible Memory (RAM)

4. Information is permanently stored in many microcomputers on which of the following: (circle letter)

a. "floppy diskettes"
b. hard disks
c. audio cassette tape
d. all of the above
e. none of the above

5. Which of the following is generally not true about caring for diskettes? (circle letter)

   a. they should be dusted frequently
   b. they should never be bent or folded
   c. they should be kept in a dust-free place
   d. they should not be placed on or near a magnetic field (e.g. radios, transformers, metallic surfaces)

6. A diskette which contains the software programs that activate a disk operating system is called a:

   _____

7. The diskette on which data is stored is usually called a:

   _____

8. The slot or peripheral box(es) in which an operator places a diskette for operation of the machine or for storage of data is called the:

   _____

9. "SuperPILOT" is: (circle letter)

   a. programmed in PASCAL
   b. a software program
   c. a lesson authoring system
   d. all of the above
   e. none of the above

10. "Primitive instructions which control the operations of a computer" describes which of the following languages: (circle letter)

    a. FORTRAN
    b. COBOL
    c. MACHINE LANGUAGE
    d. BASIC
    e. LOGO

11. Match the following:

High-level languages	a. COBOL b. FORTRAN c. BASIC
Low-level languages	d. ASSEMBLY e. PASCAL f. MACHINE

12. A device that allows transmission of information through telephone lines between two small computers or between a terminal and a large mainframe computer is called a: (circle letter)

    a. teleprompter
    b. hard disk
    c. modem
    d. telewriter
    e. CRT

13. An arrangement whereby a computer is configured to enable many users to use it at the same time with each person at a separate terminal is called a: (circle letter)

    a. time sharing system
    b. a CRT
    c. a batch mode
    d. none of the above
    e. all of the above

14. Equipment that is external to the computer such as printers, graphics board, cassettes, and disk drives are referred to as: (circle letter)

    a. CRT's
    b. interactive networks
    c. external ports
    d. peripherals
    e. none of the above

15. A software system that controls microcomputer operations is called: (circle letter)

    a. APPLEDOS
    b. TRSDOS
    c. CP/M
    d. all of the above
    e. none of the above

16. If a friend of yours wanted to buy a microcomputer that could "interface" with a number of commercial software programs, you would advise him or her to do which of the following? (circle letter)

    a. find a machine that had a lot of memory storage
    b. consider a machine that had CP/M-directed operations
    c. look at the APPLE's
    d. go to a hard disk

17. The term "external port" refers to: (circle letter)

    a. points away from the microprocessor that contain the memory
    b. locations on the machine to which peripherals can be attached
    c. large computers that receive and transmit data to and from remote sites
    d. locations on a machine from which data is unloaded
    e. all of the above

18. Most software programs for microcomputers are written in some form of which of the following languages? (circle letter)

    a. PASCAL
    b. FORTRAN
    c. COBOL
    d. LOGO
    e. BASIC

19. A "mainframe" is a general term referring to: (circle letter)

    a. large computers
    b. a minicomputer
    c. a microcomputer
    d. a personal computer
    e. none of the above

20. TRSDOS translated means: _____

    _____

21. Circle the letter next to the following variables one needs to consider in purchasing a microcomputer:

    a. color graphics
    b. memory
    c. storage equipment available
    d. the software that will run on it
    e. number of disk drives
    f. cost
    g. use
    h. availability of service
    i. type of peripherals to be to be used
    j. needs of users
    k. manufacturer's reputation
    l. service record
    m. skill of service personnel

22. "Word Handler" is: (circle letter)

    a. a word processing program
    b. a software program
    c. not programmed in BASIC
    d. all of the above
    e. none of the above

23. "BASIC" is: (circle letter)

a. a set of computer instructions
b. a programming language
c. stored in ROM on most machines
d. all of the above
e. none of the above

24. "48K RAM" means: (circle letter)

a. the machine processes information at 48,000 Kilobits a second
b. the machine has a user accessible memory storage bank equal to 48,000 bytes of information
c. the machine can randomly access 48,000 bytes of information per minute
d. all of the above
e. none of the above

25. Typing "DIR" <ENTER> on a TRS microcomputer does which of the following things? (circle letter)

a. gives the user a list of system and user files
b. turns the machine off
c. calls up the word processing information
d. allows the user to store information
e. tells the user how the memory is stored on a diskette

To edit the instructions contained in a program written in SuperPILOT, which of the four editors would you enter from the main menu? _____

_____

****

ATTITUDES: COMPUTERS IN EDUCATION

Bob Eckert, Sharon Goh, and Steve Bergman
Center for Innovation in Teaching the Handicapped
Indiana University, Bloomington, IN
May 1984

****

GENERAL Information

A. What are the LAST 6 DIGITS of your Social Security Number? (YOU WILL REMAIN ANONYMOUS)

__ __ __ __ __ __

B. Are you currently a student? (Please check ONLY one)
- [ ] 1. No, I am not currently a student
- [ ] 2. Yes, Parttime Undergraduate
- [ ] 3. Yes, Fulltime Undergraduate
- [ ] 4. Yes, Parttime Graduate Student
- [ ] 5. Yes, Fulltime Graduate Student
- [ ] 6. Other: _____

C. Are you currently a teacher? (Please check ONLY one)
- [ ] 1. No, I am not currently a teacher
- [ ] 2. Yes, Parttime Teacher in Sp. Ed. (Self-contained)
- [ ] 3. Yes, Parttime Teacher in Sp. Ed. (Resource)
- [ ] 4. Yes, Fulltime Teacher in Sp. Ed. (Self-contained)
- [ ] 5. Yes, Fulltime Teacher in Sp. Ed. (Resource)
- [ ] 6. Yes, Parttime Teacher in Reg. Ed.
- [ ] 7. Yes, Fulltime Teacher in Reg. Ed.
- [ ] 8. Other: _____

D. What is your highest degree?
- [ ] 1. High school diploma
- [ ] 2. Bachelors
- [ ] 3. Masters
- [ ] 4. Doctorate

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

E. How much training (in clock hours) have you received in the use of microcomputers in the classroom?
(If you have received NO training, please enter a 0 (zero))

_____ hour(s)

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

****

IMPORTANT

IF YOU **HAVE NEVER BEEN** A **TEACHER**, PLEASE SKIP TO PAGE 4.
IF YOU **ARE** (OR **HAVE BEEN**) A TEACHER, PLEASE ANSWER THE FOLLOWING QUESTIONS, AND THEN PROCEED TO PAGE 5.

****

F. At what level(s) do you teach?

[ ] 1. Elementary
[ ] 2. Middle school/junior high
[ ] 3. High School
[ ] 4. Middle school/junior high and high school
[ ] 5. All of the above

G. How many years have you taught?

_____ year(s)

H. How many students make up your total case load this year (or the last year you taught)?

_____ student(s)

I. Is there a computer resource person you can go to with questions, problems, ideas? (Check as many as apply)

[ ] 1. Yes, in my school
[ ] 2. Yes, in my school system
[ ] 3. Yes, outside of my school system
[ ] 4. No
[ ] 5. I don't know

J. Do you have access to microcomputers in the following locations?

Yes[ ] No[ ] 1. In my classroom
Yes[ ] No[ ] 2. In the school's computer lab
Yes[ ] No[ ] 3. In the school's office
Yes[ ] No[ ] 4. In my home
Yes[ ] No[ ] 5. Other: _____

K. How many microcomputers do you have access to? (if NONE, please enter a 0)

_____ microcomputer(s)

L. Please indicate whether or not you use a microcomputer for the following instructional purposes:

Yes[ ] No[ ] 1. Remediation in basic skills (reading, math, etc.)
Yes[ ] No[ ] 2. To introduce of new concepts

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

Yes [ ] No[ ] 3. For curriculum enrichment
Yes [ ] No[ ] 4. For computer literacy/programming training
Yes [ ] No[ ] 5. To reinforce good performance/behavior
Yes [ ] No[ ] 6. For routine drill/practice on curriculum content
Yes [ ] No[ ] 7. For student performance/attendance record-keeping
Yes [ ] No[ ] 8. For IEP development/management
Yes [ ] No[ ] 9. To prepare reports
Yes [ ] No[ ] 10. Other: _____

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

M. How often do you **personally** use a microcomputer for any of the purposes that you checked above?

[ ] 1. Daily
[ ] 2. 2-4 times per week
[ ] 3. Once per week
[ ] 4. Once per month
[ ] 5. Less than once per month
[ ] 6. Never

N. Please indicate if you use the following types of computer programs for instructional purposes:

Yes[ ] No[ ] 1. Self-written programs
Yes[ ] No[ ] 2. Commercially available programs
Yes[ ] No[ ] 3. In-house/school system prepared programs
Yes[ ] No[ ] 4. Other: _____

O. Please estimate the average amount of time per day that each of your students uses a microcomputer under your direction over the past year (or the last year you taught):
(If **NONE** of your students use a microcomputer under your direction, please enter a 0 (zero))

_____ minute(s)

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

ATTITUDINAL Information

Please indicate your degree of agreement or disagreement with the following statements by circling the appropriate number on the scale provided to the right of each statement.

	Strongly Agree					Strongly Disagree
1. Having a microcomputer in the classroom is not a distraction to most students.	1	2	3	4	5	6
2. I am pleased with the variety of programs available for use with special education students.	1	2	3	4	5	6
3. Errors made by computers could have disastrous effects on our society.	1	2	3	4	5	6
4. Computer-based instruction motivates students to learn material in which they wouldn't ordinarily be interested.	1	2	3	4	5	6
5. There is a wide range of effective applications of computer-based instruction in the classroom.	1	2	3	4	5	6
6. I am confident of my ability to learn to use microcomputers effectively in the classroom.	1	2	3	4	5	6
7. Using computers in the classroom prepares students to meet the demands of their daily lives in the future.	1	2	3	4	5	6
8. The use of computers stimulates communication among individuals in our society.	1	2	3	4	5	6
9. Microcomputers permit me to present instruction in ways I cannot now.	1	2	3	4	5	6
10. All students should be taught to	1	2	3	4	5	6

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

program computers.

11.	Most of the software programs available are easy for students to use without help from the teacher.	1	2	3	4	5	6

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

		Strongly Agree					Strongly Disagree
12.	The use of computers enhances the daily lives of individuals in our society.	1	2	3	4	5	6
13.	I am confident I will receive (or did receive) adequate training before and during my initial use of the microcomputer in my classroom.	1	2	3	4	5	6
14.	Computers are generally too complicated for use by the average person.	1	2	3	4	5	6
15.	The educational benefits of computers outweigh the disadvantages of their use in the schools.	1	2	3	4	5	6
16.	Using microcomputers in the classroom enhances students' interpersonal skills.	1	2	3	4	5	6
17.	Most available instructional programs are age-appropriate for use with my students.	1	2	3	4	5	6
18.	I am not worried that someone will steal or vandalize a microcomputer that I am in charge of.	1	2	3	4	5	6
19.	Computer-based instruction has the potential of being more effective than traditional methods.	1	2	3	4	5	6
20.	Computers provide an instructional alternative for students who have difficulty in learning material presented through traditional methods.	1	2	3	4	5	6
21.	I can write microcomputer programs to meet the special needs of my students.	1	2	3	4	5	6
22.	The available instructional programs	1	2	3	4	5	6

Please go on to the NEXT PAGE...

Attitudes: Computers in Education

are adequately sequenced for my students.

23.	The available instructional programs adequately branch students to higher and lower instructional levels when necessary.	1	2	3	4	5	6

		Strongly Agree					Strongly Disagree
24.	Classroom use of microcomputers lessens my instructional duties.	1	2	3	4	5	6
25.	The available instructional programs are appropriate for most of my students' ability levels.	1	2	3	4	5	6
26.	Humans have sufficient control over computers to prevent ill effects of their use in our society.	1	2	3	4	5	6
27.	I am not afraid that I will damage the microcomputer or its programming.	1	2	3	4	5	6
28.	The wide use of computers in our society will create many new jobs.	1	2	3	4	5	6
29.	The average individual can learn to use computers effectively.	1	2	3	4	5	6
30.	I am confident I can choose appropriate microcomputer programs for my students.	1	2	3	4	5	6

Thank you for participati in this survey

Please go on to the NEXT PAGE...

MICROCOMPUTERS IN SPECIAL EDUCATION

MODULE 2: Software Applications

Many new words have entered the English language as a result of advances in computer technology over the past 40 years; other existing words have taken on new meanings. Knowledge of computer terminology aids not only in understanding the operation of a microcomputer but in communicating with other computer users. In addition to terminology, the microcomputer user should be aware of the multitude of applications for which computer software has been developed. Three main categories of applications software are available: home or personal, business, and education. Awareness of the range of educational applications is especially important for the teacher interested in selecting and operating software in the classroom.

Objectives

At the end of this module, participants will be able to:

1. Define and operationally use selected computer terminology related to microcomputers.
2. Distinguish between the two types of software: systems and applications.
3. Distinguish between and give examples of the various types of educational software applications.
4. Identify the three major sources of educational software.
5. Operate a microcomputer to run an educational software program.

MICROCOMPUTER TERMINOLOGY

Many new words have entered the English language as a result of advances in computer technology over the past 40 years; other existing words have taken on new meanings. A number of terms which are important to the understanding of microcomputer functioning will be discussed here. Others will be introduced in the following modules of this series.

1. MEMORY: A microcomputer's internal memory consists of a single or multiple microchips in which data or instructions are held for immediate or later use. The data or instructions are represented as a series of on/off electronic signals stored in the memory chip.

2. BIT: A bit is the smallest piece of information that can be processed by the computer and consists of one on or off electronic signal. One bit, by itself, does not convey any meaningful information. It is only when a number of bits are taken as a unit that they represent a piece of information. This piece of information is stored in a BYTE.

3. BYTE: Bytes are made up of a series of bits which form a code representing a letter, number, or symbol (any one of these is called a 'character'). Many microcomputers currently use 8 bits to form a byte (although some newer ones use 16 or 32-bit bytes). It takes 8 bits or 1 byte of memory to store one character. Memory is expressed in terms of thousands (K's) of bytes. One K equals 1,024 bytes; a 48K microcomputer can store 49,152 (48 X 1,024) bytes of information in internal memory at one time. This is equivalent to 49,152 values which make up the data or instructions to be worked with. The following illustration helps one to conceptualize memory size:

It takes 1 byte of internal memory to store one letter, number, or symbol (character).

A typewritten page holds 250 words (1,500 characters and spaces). It would take 1,500 bytes of computer space to store a page of text.

Therefore it takes:
* 1.5K of computer space to hold one typewritten page.
* 30K of computer space to hold 20 typewritten pages.
* 150K of computer space to hold 100 typewritten pages.

(Adapted from Guide to Personal Computing, 1982. Maynard, MA: Digital Equipment Corporation, p. 48.)

Of, course, not all information entered into the computer will be text. Some will be instructions in the form of a computer program and some will be raw data to be manipulated in some way by a program. Memory is a term not only describing a computer's

internal storage; it also describes external storage on a diskette or cassette and the terms bit, byte, and 'K' take on the same meaning in this context.

4. ROM: This term stands for Read Only Memory. This is a permanent part of the computer's internal memory which is preprogrammed by the manufacturer and which cannot normally be altered. ROM usually contains the computer's essential instructions which allow the CPU to communicate with the input and output peripherals. For example, when the computer is turned on, the CPU uses instructions stored in ROM to access the disk drive for further instructions to act upon.

5. RAM: RAM stands for Random Access Memory. Unlike ROM, RAM can be programmed, edited, or written over. RAM is the portion of internal memory which holds the programs which are being operated as well as the data or other information which are being manipulated by the programs. If disk drive units are being used, the program on the diskette in the drive unit will be transferred into the computer's RAM and executed by the CPU. Any information entered from an immediate communication device such as a keyboard or joystick is also entered into RAM. Subsequent programs or information are written over previous ones in RAM so that, while RAM is never cleared, the full amount of storage space is available for each new program.

6. PROM: Some microcomputers provide a space in internal memory called PROM, which means Programmable Read Only Memory. This is a special section of memory into which the user can insert his or her own permanent instructions for the CPU. PROM is used for custom programs such as a routine to communicate with a special peripheral not normally connected to that computer (Taber, 1983). PROM (Erasable Programmable Read Only Memory) is also provided in some microcomputers. Unlike ROM and PROM, programs in EPROM can be erased by electronic means and then reprogrammed. EPROM is often used by programmers in the development and testing of programs which are to be permanently placed into PROM.

7. DISK OPERATING SYSTEM (DOS): The DOS is the master program which every microcomputer needs to manage transfer of information to and from and storage on a diskette. The major functions of the DOS include: 1) organization of the huge pool of bytes on the diskette into a meaningful format for the user (this takes the form of a group of files which may contain data or programs, as well as a directory or catalog of those files); 2) management of loading into internal memory and initial execution of a program stored on a diskette; 3) retrieval and storage of file contents from internal memory to the diskette and vice versa; 4) provision of an interface from a running program to information drawn from, modified on, or written to a diskette; and 5) erasure of file contents on a diskette. These functions can be accessed either by a request from another, running, program, or by direct request from the user at the keyboard.

   The DOS is sometimes stored in ROM, but more often is stored on a diskette. It is important to remember that the DOS is not part of

the computer's hardware, but a software program which provides operating instructions. Each brand of microcomputer generally uses its own version of an operating system. For example, the IBM PC uses PCDOS, the Radio Shack TRS-80's use TRSDOS, and the APPLE's use APDOS. This lack of a standard disk operating system means that, generally, software written for one brand of computer often cannot be used on another brand. However, Digital Research Corporation has developed an operating system called CP/M which enables many different brands of microcomputers to share software.

8. BOOT: This term means to ready a computer for use by loading the disk operating system into the computer's temporary memory, or RAM. When the computer is turned on initially or reset after some use, booting is generally automatic. The computer's ROM instructions allow the CPU to communicate with the disk drive unit in order to load the DOS into RAM. When this process is completed, the computer will notify the user that the DOS is functioning and ready for further directions.

9. SYSTEM DISK: Diskettes on which programs are stored are called system disks. These disks also usually contain the DOS program. When the computer is turned on, the DOS program is automatically loaded from the system disk into RAM in internal memory. The DOS then manages the execution of the other programs stored on the system disk.

10. DOCUMENT DISK: A document disk is one on which data is stored. It usually does not contain programs and may be erased for reuse. For example, some educational software is designed to collect data on student performance while using an instructional program. A system disk may contain the program which provides the instruction, but during or at the end of the learning session, information on the student's performance would be written to a document disk for storage. In a microcomputer with two disk drives, the system disk would be placed in one drive and the document disk in the other. In computers with only one drive the two types of disks would be inserted and removed in turn at appropriate points in the program.

    There is no intrinsic difference between the physical layout of system and document disks. Any blank disk will serve either purpose and, indeed, both programs and data can be stored on the same disk. For relatively simple programs this provides no problem. However, for lengthy programs which use much disk space, a separate document disk may be more appropriate. Other convenience and confidentiality factors may also dictate use of separate document disks.

SOFTWARE TYPES, APPLICATIONS, AND SOURCES

Types and Applications

There are two basic types of computer software, systems software and applications software. Systems software includes those programs which instruct the computer hardware how to perform its operations. These programs include:

* Operating system software: programs which monitor and control the running of all other software.

* Assemblers, compilers, and interpreters: programs which translate programs written in high-level languages to a language directly readable by computers.

* Peripherals operating software: programs which manage instructions from the CPU to the peripherals connected to the computer.

* Debuggers: programs which help the computer programmer locate errors in other programs.

* Editing system software: programs which allow the user to correct errors in text, data, or instructions being entered into the computer system.

Systems software is used primarily by computer programmers. Most educators, unless they are creating their own sophisticated programs, will use applications software.

Applications software is designed to perform a specific task such as presenting an instructional program on English grammar or statistically analyzing data collected by a researcher. There are three general categories of applications software: home, business, and educational.

1. HOME Applications: Microcomputers are used in the home for many purposes. Some of these include budgeting and record-keeping of income and expenses for personal or tax purposes, inventories of personal belongings and home contents, continuing education through instructional programs, correspondence through word-processing and mailing list management programs, and for entertainment. Personal computers are also used in conjunction with a modem and a regular telephone to provide access to the rapidly growing number of information databases and consumer services such as stock market news, bank account management, shopping services, and travel information among many others.

2. BUSINESS Applications: Many of the functions carried out in the average business office are now being handled more efficiently by microcomputers. Business software applications include word processing for production of letters, reports, and any other text-related materials; data processing for sorting, summarizing, or recording numerical data; statistical evaluation for analyzing product quality, consumer characteristics, etc.; market forecasting to analyze and project consumer trends; and inventory

control for maintaining efficient and up-to-date inventory records.

3. EDUCATIONAL Applications: Every teacher currently active in the profession is by now aware that microcomputers are rapidly appearing in most school buildings if not most individual classrooms. Instructional applications of microcomputer software include:

A. Drill and Practice. Most of the instructional software available today can be classified as drill and practice programs. These programs take advantage of the computer's ability to endlessly and patiently present exercises designed to help the student master a specific skill. Randomly generated exercises are displayed, the student enters an answer, and the computer provides feedback on that answer. Drill and practice software is available for reinforcement of basic math skills, spelling skills, vocabulary development, foreign language vocabulary, history dates and periods, teaching chemistry, music, and geography facts, and nearly any other information or skill taught in a conventional curriculum. The immediate feedback and non-judgmental attitude present in computer drill and practice activities encourage repetition by the learner. The "patience" inherent in this type of instruction allows the learner to progress at his/her own speed and frees the teacher from some time-consuming and tedious tasks. A disadvantage of simple drill and practice software is that it can become boring to the learner after only a few uses of the same program. Some programs combat this by presenting the exercises in a game format, adding audio and visual effects, using the learner's name in messages displayed, and/or varying the feedback provided after each exercise.

B. Tutorial. Programs which are designed to teach concepts are called tutorials. These programs present curriculum materials in a programmed learning format and branch the student to higher or lower levels based on the student's mastery of the concept. Tutorial programs can be used either to re-teach coursework previously presented but not mastered, or to introduce new concepts or topics. Most tutorials include pre- and posttests as well as intermediate "probes" to assess the learner's functioning levels. Many also incorporate drill and practice exercises to aid in mastery of the concepts presented. Tutorial software is difficult to produce because it must accommodate different student ability levels and learning styles. For this reason, good tutorial software is not easily found and the tutorial software that is available commercially must be carefully evaluated before purchase or use.

C. Simulations. A great advantage of computers is that they may be used to recreate real-life situations that are not economical, convenient, or even possible to reproduce otherwise in the traditional classroom. These re-creations are called simulations and attempt to model the set of conditions found in a real phenomenon. The purpose of a simulation may be simply to allow students to experience this set of conditions or to allow them to make choices and then deal with the consequences of these choices. Situations such as chemistry experiments, space travel, historical

events, driving a car, and human body functioning can all be presented by simulation software. While these programs hold great promise for education, they are also difficult to produce and, thus, limited in availability at this point.

D. Problem Solving. Programs designed to teach or utilize logic and analytical skills are classified as problem-solving. The student inputs, manipulates, analyzes, or compares relatively unstructured information to solve a problem or produce a final product. Creative writing through word processing software is one example of a problem-solving application. Others include Science or Social Studies research, comparison or analysis of literary works in English, equipment design or analysis in Vocational areas, and computer programming itself.

E. Instructional Planning. A variety of programs are now available which aide the teacher in planning instruction on an individual or group basis. For special education, this software most often provides a means of developing and monitoring individualized education programs (IEP's). These programs often contain libraries of curricular goals and objectives from which the teacher may choose for individual students. The chosen goals and objectives are then stored on the students' IEP's and a means is provided for each IEP to be updated at regular periods. Instructional planning software may also store lists of teaching materials and strategies keyed to specific goals and objectives, assessment instruments to measure student progress toward meeting the goals and objectives, and a means of providing individual or group reports on a periodic basis.

F. Computer-Managed Instruction. Computer-managed instruction (CMI) incorporates the learning prescription and record-keeping components of instructional planning software with thorough testing and diagnosis capabilities. It is used to measure the student's present level of performance in one or more subject areas, prescribe an individual education program for the student, direct the student to the appropriate assignments (computer programs, textbooks, worksheets, etc.), monitor the student's progress, and provide periodic reports on an individual or group basis. These functions relieve the teacher of many time-consuming and laborious tasks and allow him/her to spend more time in helping students master curricular content. While computer-managed instruction has been shown to result in significant educational gains, effective CMI software is expensive and difficult to produce because of its complexity. In addition, since curriculum content and requirements differ widely among school districts, school buildings and individual classrooms, a CMI package found effective by one teacher for his/her students may not be appropriate for another classroom. Thus, CMI software, like all other applications software, must be thoroughly evaluated by the potential user.

G. Administration and Record-Keeping. Other educational software applications of interest to teachers relate to non-teaching tasks which are necessary but time-consuming and laborious. These include maintaining records related to student assessment, attendance, scheduling, and grading information; maintaining

equipment and materials inventories; generating reports to students, parents, and administrators; and data collection for evaluation of teaching materials, techniques and strategies. Software applications for school administrators are also many and varied. They include maintenance of and instant access to records related to student assessment data, student cumulative files, attendance, student schedules, school calendars, library holdings, classroom or building inventories, school census data, financial information, supply requisitions, and countless other administrative tasks.

This is only a partial listing of the educational applications software currently available. It should be noted that some software will incorporate two or more of the applications described above. For example, a math program may combine drill and practice with concept presentation and record-keeping of student performance. A tutorial program may also provide problem-solving and simulation activities.

It is important to remember that any software being considered should be thoroughly evaluated to insure that it meets the needs of the potential user(s). Unfortunately, while an overwhelming amount of educational software is commercially available, many programs are unsuitable for classroom use, particularly with special needs students. Some programs rely heavily on reading skills which the special student may not possess. Others provide no, or inappropriate, feedback to students and make no provision for recording student performance data. Others do not allow the teacher to vary content or presentation parameters in order to individualize for different students. And some programs require too much teacher assistance because directions are too complex, inadequate, or simply nonexistent. Additional problems include unmotivating formats, undiscovered "bugs" which cause the program to fail, and poor documentation or user's guides to aid in implementation of the software.

High-quality educational software <u>is</u> available. When well-written and field-tested programs are used, microcomputers possess great potential for enhancing student achievement and for simplifying teachers' non-instructional tasks. <u>Effective</u> software enables the computer to:

* Motivate and reinforce students with histories of academic failure.

* Provide instant feedback and automatic branching to appropriate levels within the program.

* Allow the learner to progress at his/her own rate.

* Provide information on an individual basis.

* Provide enrichment material within the curriculum.

* Automatically monitor and record student performance.

* Decrease teacher time spent on laborious and tedious "housekeeping" tasks.

* Provide administrators with instant access to up-to-date information on student and teacher data.

Sources

There are three basic sources of educational software: commercial software available through catalogs or computer stores, self-developed software using a programming language such as BASIC or PASCAL, and self-developed software using lesson authoring systems which allow the developer to insert desired content and choose among options from a pre-determined program format.

There are literally hundreds of publishers and distributors of commercial software across the country. Some microcomputer manufacturers such as IBM, APPLE, and Radio Shack employ their own programmers to develop software which is made available through company catalogs or distribution centers. There are also many software development corporations which design programs for various brands of microcomputers and distribute them through catalogs and regional centers. Finally, there are many strictly distribution companies that purchase and make available software produced by computer manufacturers, software development companies, and private individuals. Lists of software distributors are often found in recently published books on computers in education. Local computer stores and the many popular computing magazines are also good sources for information on software.

The individual proficient in programming can, of course, produce original software designed to meet the needs of specific users. Popular programming languages such as BASIC, PASCAL, and FORTRAN are taught at local technical schools, universities, adult education classes, and some computer stores. Motivated individuals can learn programming independently from textbooks or tutorial software available for most microcomputers. While highly desirable for maximum classroom usability, self-programmed software is time-consuming and tedious to produce.

An alternative to original programs involves the use of lesson authoring software which allows non-programmers to develop programs tailored to individual student needs. The simplest authoring systems provide a lesson skeleton with mode of presentation already programmed. The teacher need only type in the curricular content to be presented. More complex authoring systems involve a set of commands which allow the teacher to not only enter the curricular content, but also to select presentation formats, presentation mode, and other options. Since curricular content and content management and presentation can be changed at will, good authoring systems are extremely versatile and eliminate the need for repeated purchases of commercial software. Simple authoring systems can be learned in a few hours; more complex systems will, of course, take more time. Development of a lesson can take from minutes to several hours or longer depending upon the complexity of the authoring system and/or lesson and on familiarity with the system. A later module in this series will provide an introduction to and practice using a versatile authoring system called APPLE SuperPILOT.

Table 2. Educational Applications of Microcomputer Software

APPLICATION	DESCRIPTION
Drill and Practice	Question and answer formats presented to the student on the microcomputer. Used to reinforce information previously introduced and to provide practice.
Tutorial	Presentation of concepts as well as drill/practice on skills involved. Used to initially teach a concept or to present the concept in a new and different manner. Usually includes assessment of student's understanding of the concept.
Simulations	Recreation of a real-life situation, often with graphic and audio effects. Used to teach decision-making and to help students deal with consequences of their decisions.
Problem Solving	Programs designed to stimulate students to use logic to solve a problem. These programs usually involve analytic, comparison, and other transferred skills.
Instructional Planning	Programs designed to aid the teacher in developing and monitoring IEP's or individualized programs for students. These programs may store curricular goals and objectives, lists of teaching materials and strategies, and assessment items to measure student progress.
Computer-Managed Instruction	Multi-faceted programs which assess students' current level of performance, select appropriate curricular content, present context or direct student to non-computer curricular sources, and monitor individual or group progress.
Administration and Record-keeping	Clerical programs which aid teachers and administrators in calculating and recording grades, storing student records, and compiling student data for local, state, and federal purposes.

Module 2

Transparency #1

MICROCOMPUTER TERMINOLOGY

MEMORY

BIT

BYTE

ROM

RAM

PROM

DISK OPERATING SYSTEM (DOS)

BOOT

SYSTEM DISK

DOCUMENT DISK

Module 2

Transparency #2

TYPES OF COMPUTER SOFTWARE

A. SYSTEMS SOFTWARE

1. Operating System Software
2. Assemblers, Compilers, and Interpreters
3. Peripherals Operating Software
4. Debuggers
5. Editing System Software

B. APPLICATIONS SOFTWARE

1. Home Applications
   - -Budgeting
   - -Record-keeping
   - -Inventory
   - -Continuing Education
   - -Correspondence
   - -Entertainment

2. Business Applications
   - -Word Processing
   - -Data Processing
   - -Statistical Evaluation
   - -Market Forecasting
   - -Inventory Control

3. Educational Applications
   - -Drill and Practice
   - -Tutorial
   - -Simulations
   - -Problem-Solving
   - -Instructional Planning
   - -Computer-Managed Instruction
   - -Administration and Record-keeping

Module 2

Transparency #3

ADVANTAGES OF MICROCOMPUTER USE IN THE CLASSROOM

1. Motivates and reinforces students
2. Provides instant feedback and automatic branching
3. Allows learner to progress at own rate
4. Allows individualized instruction
5. Provides enrichment material
6. Monitors and records student performance automatically
7. Decreases teacher time spent on record-keeping tasks
8. Provides administrators instant access to up-to-date records

CMMRS
COMPUTER-MANAGED MATH REMEDIATION SYSTEM
VERSION 2.0

Software Development: Robert Eckert (Version 2.0)
George J. Haus (Version 1.0)

INTRODUCTION

The Computer-Managed Math Remediation System (CMMRS) is a software system comprised of four programs designed to: (1) acclimate the student to using numbers on the computer keyboard (DIGITS program), (2) assess students' current performance in each of the four basic math operations (MATH program), (3) systematically probe and remediate student performance through 16 groups of math skills (MONITOR program), and (4) report student performance data in each of the three preceeding programs (REPORT program).

HARDWARE REQUIREMENTS

CMMRS is developed for the TRS-80 Model I and III/4 (48K, dual disk drive).

SOFTWARE REQUIREMENTS

CMMRS consists of a program disk - which contains the four programs, and a student data disk - which you prepare with the FORMAT program provided by the TRS-80's operating system (TRSDOS). Each student data disk will hold records for 16 students.

OPERATION

Running CMMRS

(a) Turn on the machine.
(b) Insert the program disk in drive 0 and the student data disk in drive 1.
(c) Press the reset button.
(d) From the TRSDOS Ready prompt, you may access one of the four programs available through CMMRS - DIGITS, MATH, MONITOR, and REPORT. Each of these programs is described below.

DIGITS Program

CMMRS programs use speed of response as one measure of student performance. Yet, some students may not have extensive experience in entering numeric data through a keyboard, and therefore slow performance may be due to unfamiliarity with the keyboard rather than mathematical difficulties. To address this problem, the DIGITS program was developed to assess the proficiency with which a student can use the numeric keyboard. Student proficiency is measured in correct digits per minute (CDM) and error digits

per minute (EDM). To run the DIGITS program, type Basic Digits <enter> after receiving the TRSDOS Ready prompt. DIGITS will present the student with five 1-minute drills of matching numbers. In order to collect enough data to determine if a student is experiencing difficulty with the keyboard, it is suggested that the student attempt 2 to 3 sessions with the DIGITS program.

MATH program

The MATH program is similar to a pre-test; it provides an initial assessment of the student's performance in basic addition, subtraction, multiplication, and division. This information is important for setting the goal criteria for the MONITOR program (which administers probes and remediation). To run the MATH program, type Basic Math <enter> after receiving the TRSDOS Ready prompt. The MATH program will then present the student with four 2-minute drills in each of the basic math operations, with an even distribution of problem types in each area. After all four trials have been completed, the MATH program will display a record of student performance in each of the four areas, calculated in correct digits per minute. In order to collect enough data to set goal criteria for subsequent remediation, it is suggested that 3 or 4 sessions of the MATH program be administered to each student. Also, the MATH program should be used periodically as a reassessment device.

MONITOR program

The MONITOR program administers a series of probes to the student, and on the basis of these probes, assigns appropriate remedial activities. Before running the MONITOR program, the goal criteria and program operation options should be set through the REPORT program (described below). To access the MONITOR program, type Basic Monitor <enter> after receiving the TRSDOS Ready prompt. The MONITOR program will then administer a 5 to 6 minute session that consists of probes and appropriate remediations. NOTE: To exit the DIGITS, MATH, or MONITOR program, type <enter> when prompted for a student's name OR type "exit" <enter> after the program is in progress.

REPORT program

The REPORT program is accessed by typing Basic Report in response to the TRSDOS Ready command. Once accessed, the REPORT program will present you with a menu as follows:

<1> REPORT DIGITS Assessment Program DATA
<2> REPORT MATH Assessment Program DATA
<3> REPORT MONITOR Probe/Remediate Program DATA
<4> ACCESS MONITOR Achievement CRITERIA
<5> ACCESS Student NAMES/NUMBERS File
<6> HELP
<7> EXIT

The first command, report DIGITS assessment program data, allows you to view a student's performance on trials of the DIGITS program. Likewise, the second and third commands, report MATH assessment program data and report MONITOR probe/remediate data, permit you to view records of student performance from MONITOR probe and remediation sessions. The fourth command,

access MONITOR achievement criteria, permits you to view and/or alter the goal criteria to be used during the MONITOR program. The fifth command, access student names/numbers, enables you to assign, change, delete, or view student names and numbers located on the student data disk. For the five commands described above, you may view the information you request on the screen or you may obtain a printed copy. The sixth command, help, informs you how to use the REPORT program, and the seventh command, exit, returns you to TRSDOS.

CIRIS
Computer-Based Informal Reading Inventory System
Version 2
for the
TRS-80 Model III/4/4P (using TRSDOS 1.3)

Software Development. Robert Eckart (Version 1 and 2)

INTRODUCTION

The Computer-Based Informal Reading Inventory System (CIRIS) consists of several programs designed to administer silent reading passages to a student and to assess his/her reading comprehension of these passages. Stories 1 through 10 are provided with the program disk; these stories are appropriate for reading levels grade one through three and their content and questions are not alterable. Stories 11 through 20 are available for your use, and CIRIS has been designed to accept passages entered and analyzed by CRIS sofware and to convert these into CIRIS stories.

CIRIS includes extensive diagnostic and record-keeping capabilities. CIRIS utilities enable you to access student data and produce a report of student progress to date. You can also change criteria associated with reading speed and comprehension, the provision of feedback, and the opportunity to 'peek' back at the story while answering comprehension questions.

HARDWARE REQUIREMENTS

CIRIS is developed for the TRS-80 III/4/4P (48K, 2 disk drives).

SOFTWARE REQUIREMENTS

CIRIS consists of a program disk - which stores all the system programs, stories and comprehension questions, and a student data disk - which you prepare with the FORMAT program provided by the TRS-80's operating system (TRSDOS). Each student data disk will hold records for 16 students.

OPERATION

Running CIRIS

(a) Turn on the machine.
(b) Insert the program disk in drive 0 and the student data disk in drive 1.
(c) Press the reset button.
(d) When you receive the **TRSDOS Ready** prompt, you may choose either the READING or the REPORT program. Type:

DO READING<ENTER>

if you wish to read a story and answer comprehension questions (student activities) or type:

DO REPORT<ENTER> if you wish to view student data or modify student performance criteria and/or CIRIS stories and questions (teacher activities). The Reading and Report programs are described below in more detail.

READING Program

The READING program presents a story to the student user. By default, automatic story selection is set to yes, and unless you change this through the report program, CIRIS automatically steps the student through the first 10 stories, advancing him/her only when both accuracy and speed meet criteria. The student reads the video display of the story that has been selected and is timed while doing so. The student then is presented with multiple-choice comprehension questions and is timed while answering these. If you desire, the student may be permitted to 'peek' back into the story for an answer, and a separate timing is kept for this scanning process.

REPORT Program

Upon typing **DO REPORT<ENTER>**, you will be presented with the following command menu:

<1> REPORT READING INVENTORY Program DATA
<2> ACCESS READING INVENTORY Achievement CRITERIA
<3> ACCESS Student NAMES/NUMBERS File
<4> ACCESS STORIES, QUESTIONS, and ANSWERS
<5> CONVERT a CRIS Passage to a CIRIS Story
<6> HELP
<7> EXIT

The first command, REPORT READING INVENTORY Program DATA, reports student data collected by the reading program.

The second command, ACCESS READING INVENTORY Achievement CRITERIA, permits you to view, alter, or print the goal criteria that is used by the reading program .

The third command, ACCESS Student NAMES/NUMBERS File, allows you to assign, change, delete, view, or print student names and numbers from the student data disk.

The fourth command, ACCESS STORIES, QUESTIONS, and ANSWERS, enables you to display and/or print the stories, questions, and answers used in the CIRIS reading program.

The fifth command, CONVERT a CRIS Passage to a CIRIS Story allows you to transfer a passage that has been created, edited, and analyzed by CRIS. The transmitted passage becomes a CIRIS story and is assigned a number from 11 to 20. Once the transfer is complete, you may create and edit your own comprehension questions and answers for the story using the fourth command described above.

The sixth command, HELP, displays information about how to use CIRIS.

Finally, the EXIT command returns you to TRSDOS.

SPELLMASTER
Computer-Based Informal Spelling Inventory System
Version 2
for the
TRS-80 Model III/4/4P (using TRSDOS 1.3)
equipped with a Votrax Type 'N Talk Speech Synthesizer

Software Development: Robert Eckert (Version 1 and 2)

INTRODUCTION

SPELLMASTER is a microcomputer-based informal spelling inventory system that may be used to teach, assess, and report student spelling skills. SPELLMASTER enables you to compile custom-made word lists to allow for maximum flexibility and individualization in spelling activities. In addition, SPELLMASTER may be used with a speech synthesizer device for audio output. SPELLMASTER includes extensive record-keeping facilities that permit you to alter system operation and achievement criteria for each student individually.

HARDWARE REQUIREMENTS

SPELLMASTER is developed for the TRS-80 Model III/4/4P (48K, 2 disk drives), and is designed to be used with a Votrax Type 'N Talk Text to Speech Synthesizer (and speaker or headphones).

SOFTWARE REQUIREMENTS

SPELLMASTER includes a program disk, on which system programs and word lists are stored, a student data disk, which holds information about the performance of up to 16 students, and a dictionary disk, which can hold 2 to 3 'dictionaries' of up to 500 words each.

OPERATION

Running SPELLMASTER

(a) Turn on the machine.
(b) Insert the program disk in drive 0 and the student data disk in drive 1.
(c) Press the reset button.
(d) If the Votrax is to be used, make sure the cable to the TRS-80, the power supply, and headphones (or speaker) are connected properly.
(e) From the **TRSDOS Ready** prompt, you may access two different programs - SPELL and REPORT. Each is described in more detail below.

SPELL Program

The SPELL program presents a spelling inventory to the student and records his/her responses on the student data disk. In the spoken mode, each word in the spelling list, assigned by you with the set criteria function of the REPORT program, is spoken to the student, along with a verbal prompt if desired. If the student wishes the word repeated, he/she simply presses <ENTER> after the spelling word prompt. In the visual mode, each word in the

assigned spelling list is presented on the screen for the number of seconds you have specified in the criterion file. MAKE CERTAIN THAT IF SPELLMASTER IS TO BE OPERATED WITHOUT A VOTRAX, THE CRITERION FILE IS SET TO VISUAL MODE. A spelling session may be stopped altogether by pressing <@><ENTER> in response to a spelling word prompt. To access the SPELL program, type: **DO SPELL<ENTER>** after receiving the **TRSDOS Ready** prompt.

REPORT Program

You may use the REPORT program to report student scores, assign students and student criteria, specify system operation options, and create and edit spelling word lists. To access the REPORT program, type **DO REPORT<ENTER>** after receiving the **TRSDOS Ready** prompt. A command menu will then be displayed which appears as follows:

<1> ASSIGN Students to Student Data Disks
<2> SET Student Achievement Criteria
<3> REPORT Student Performance Data
<4> ACCESS Spelling Word Lists and Dictionaries
<5> HELP
<UP-ARROW> EXIT

The first command, ASSIGN Students to Student Data Disks, permits you to add, edit, delete, and show student names and numbers.

The second command, SET Student Achievement Criteria, permits you to alter achievement criteria for a specified student, including the spelling list to be administered, required accuracy criteria, the mode of presentation (visual or spoken), the duration of visual prompts, and the type and location of feedback within the spelling session.

The third command, REPORT Student Performance Data, provides you with a report of student performance for each spelling session, including spelling lists administered, correct and incorrect performances, and response times.

In order to create the spelling lists used by students in the SPELL program, you must first create a dictionary of words and then assign dictionary words to individual spelling lists. The fourth command, ACCESS Spelling Word Lists and Dictionaries, permits you to create and modify dictionaries and word lists with functions for adding, editing, viewing, and deleting individual words and for exchanging and cataloging word lists.

The fifth command, HELP, provides you with information about how to use the REPORT program.

Finally, the EXIT command returns you to TRSDOS.

CRIS
COMPUTER-BASED READABILITY INDEX SYSTEM
VERSION 3.0

Software Development: Robert Eckert
George J. Haus (Versions 1.0 and 2.0)

INTRODUCTION

The Computer-Based Readability Index System (CRIS) is a microcomputer-based software package for the determination of readability levels of educational reading materials. CRIS can analyze a reading passage of up to 1600 characters in length in four minutes, using one of three readability indexes: (1) the Spache, which is appropriate for reading materials from kindergarten through third grade, (2) the Harris-Jacobson, which is appropriate for reading materials from kindergarten through sixth grade, and (3) the Dale-Chall, which is appropriate for reading materials from fifth grade through college. The results of the readability analysis can be shown in a video display or in a more detailed, printed form. CRIS can store up to 65 1600 character passages on a single-sided, 5 1/4" double-density disk.

HARDWARE REQUIREMENTS

CRIS is developed for the TRS-80 Model I and III/4 (48K, dual disk drive).

SOFTWARE REQUIREMENTS

CRIS consists of a program disk - which stores all the system utilities, and a passages disk - which you prepare with the FORMAT program provided by the TRS-80's operating system (TRSDOS).

OPERATION

Running CRIS

(a) Turn on the machine
(b) Insert the program disk in drive 0 and the passages disk in drive 1.
(c) Press the reset button.
(d) When you receive the prompt TRSDOS Ready, type Do CRIS <enter>.

CRIS Command Menu

The above sequence of steps should cause the CRIS command menu to appear on the screen. The command menu gives you a choice of the following functions:

<1> CREATE passage(s)
<2> DELETE passage(s)
<3> EDIT passage(s)

<4> DISPLAY passage(s)
<5> PRINT passage(s)
<6> PROCESS passage(s)
<7> EXIT

To choose a function, you press the corresponding number key. Below, each function is described briefly.

CREATE passage(s) - allows you to enter a passage for subsequent analysis. Upon choosing this function, you will be asked to supply the name of the passage you wish to create, which must be 1 to 8 characters in length and consist only of letters (A-Z) or letters with numbers (0-9). Names should not begin with a number, and CRIS will not allow you to create two passages with the same name. In order to use the readability formulas apropriately, it is important to enter at least 100 words of text in sentences when you create a passage. If you make an error while typing the passage, you may backspace to correct it or you may make corrections later from the EDIT function.

DELETE passage(s) - allows you to delete passages that you no longer wish to use or store.

EDIT passage(s) - allows you to make corrections on passages you have previously created. After you specify the name of the passage you wish to edit, CRIS will display the passage on the screen and prompt you with the following command: "Search text?" In order to correct an error, you must respond to this prompt by typing the portion of the text that contains the error. Once specified, CRIS searches for the text you have indicated, displays it to you, and prompts you for the "Replacement text", or the correct form of the text. You may make a correction that adds up to 100 additional characters to the passage. If you wish to delete the erroneous portion of the text, merely press the <enter> key in response to the "Replacement text?" prompt. When you are finished editing, press <enter> in response to the "Search text?" prompt.

DISPLAY passage(s) - allows you to display a specified passage on the screen. If you wish CRIS to pause as it displays the passage, press the <P> key several times and press the <C> key to resume the display. If you wish to stop the display before it is completed, press the <S> key.

PRINT passage(s) - allows you to print a specified passage on a printer.

PROCESS passage(s) - allows you to analyze the readability of a passage you have previously created. CRIS will ask you to specify the passage(s) to be processed and will ask you to select from among the three readability formulas. Once processing has begun, you will be asked whether or not you wish to use the printer. If you choose to use the printer, you will receive lists of words that were found and not found on the readibility word list that was used to process your passage. Both the screen display and the printed display provide information about the number of words and sentences sampled from the passage, the average sentence length, and the grade level of the passage as computed by the specified readability formula.

EXIT - allows you to leave CRIS and return to TRSDOS. ONLY AFTER YOU HAVE EXITED CRIS SHOULD YOU REMOVE THE PASSAGES AND PROGRAM DISKS FROM THE MACHINE.

MICROCOMPUTERS IN SPECIAL EDUCATION

MODULE 3: Introduction to Word Processing

Word processing is the use of software to produce text-related documents on a microcomputer. A word-processing system can do virtually anything a typewriter can do, and much more, in less time and with less effort once the system is learned. For teachers, word-processing provides a method of quickly producing and easily updating and storing testing instruments, handouts, and reports to students, parents, and other school personnel. Word processing also provides students with the means to create and easily revise compositions without the untidy results frequently produced by many special education students using the traditional pen and ink methods. It also teaches valuable typing skills and can provide many students with a marketable vocational skill.

Objectives

At the end of this module, participants will be able to:

1. List four applications of word-processing systems in the special education classroom.

2. Identify the major features of word-processing software in general.

3. Identify the major features of a specific word-processing system (Word Handler).

4. Identify the function of the major control key actions in Word Handler.

5. Demonstrate proficiency in inserting and editing text using Word Handler on an Apple microcomputer.

6. Produce a printed document prepared with Word Handler on an Apple microcomputer.

EDUCATIONAL APPLICATIONS OF WORD PROCESSING

Introduction

Word processing is the use of computer software to produce text-related documents which may be printed on paper and/or stored on diskettes or tape for later use. The electronic representation of printed text provides much more flexibility and power than available with an ordinary typewriter. All keystrokes are automatically stored in memory and displayed immediately on the computer screen. Thus, the initial draft of any document is electronic, allowing the user to manipulate characters, words, paragraphs, and even pages without retyping the entire document as would be necessary with a typewriter. This electronic "cut-and-paste" characteristic of word processing systems permits any number of revisions before the ideal configuration is printed on paper.

Advantages of Word Processing

Word processing software can do virtually anything an ordinary typewriter can do, and much more, in less time and with less effort. As mentioned above, the computer permits very rapid global changes in previously entered text, eliminating the need for retyping the entire document. Thus, word processing saves considerable time and expense through:

1. Faster composition, as authors can compose at the keyboard without worry about mistakes which would otherwise be time consuming to correct.

2. Cleaner copies, as defective text can be easily corrected and reprinted quickly.

3. Electronic document storage, eliminating the need for hard copy storage space.

4. Rapid accessibility of information, as particular points in a document can be located for editing through the use of a few simple commands.

5. Rapid merging of text from various sources through electronic "cut-and-paste" procedures.

6. Rapid transport of documents from site to site, through electronic transmission over phone lines.

7. More accurate documents, through the use of auxillary programs which check for spelling errors and irregularities in grammar punctuation, and phraseology.

8. Improvement in staff efficiency and morale.

Ideal Features of Word Processors

Word processing software is available for virtually every microcomputer on today's market. Users of the popular microcomputer brands such as Apple, IBM, Radio Shack, and Commodore will have a relatively wide range of different word processing software packages from which to choose. Unfortunately, there exists almost no standardization across packages at this point. Thus, individuals must learn a substantial amount about each new system whenever a switch between packages is made.

Despite their differences in control-key functions and procedures, good word processing packages will have certain features in common:

ELECTRONIC STORAGE of Documents: temporarily in RAM, more permanently on a diskette or cassette tape.

WRAP AROUND: a feature which allows the user to type continuously without hitting the RETURN key at the end of each line. The software senses when the line of type is nearing the right margin and automatically "wraps" the next word to the beginning of the next line.

MENUS: video displays which list options available to the user and direct him/her in the use of the options.

FOOLPROOF FEATURES: checkpoints which force the user to save documents, verify requests for deletions, and exit the program without damage to open documents.

SPECIAL FUNCTION KEYS: keys that can be programmed by the user to insert certain phrases or to perform routine and frequent operations with fewer keystrokes.

SCROLLING: automatic movement of present text upward on the screen as new text is entered.

INSERTIONS: flexibility to allow users to insert text at any point in a document.

DELETIONS: flexibility to allow users to delete character by character, word by word, line by line, or paragraph by paragraph, at any point in a document.

TEXT EXCHANGES OR MOVES: flexibility to allow users to exchange words, sentences, paragraphs, or pages with others in a document, or to move text from one location to another. This feature is also called BLOCK MOVEMENTS.

AUTOMATIC SEARCH: feature which automatically searches for and places cursor on word or phrase specified by the user, eliminating need for user to visually scan the entire document. Should be able to search forward or backward through the document.

AUTOMATIC REPLACE: automatically replaces a word or phrase with another specified phrase. Can be directed to replace one occurrence of the word or phrase, or replace all occurrences (global replace) in a document.

SCREEN FORMATTING: feature which allows the user to set parameters such as line width, screen length, tabs, upper and lower cases, page breaks, centering of lines, justified text, headers and footers, etc.

PRINTING OPTIONS: feature which allows the user to set parameters for the printed page such as type pitch, character font, underlining, bold face characters, variable margins, automatic pagination, etc.

EASE OF OPERATION: well-thought-out design which incorporates features capitalizing on:

* speed of operations.
* economical control-key commands.
* individual commands for various keys.
* logical control-key commands.
* easy maneuverability of cursor around text.

Efficient and responsive word processing software will incorporate all these features. However, each package will differ in the specific user commands to perform each function. For example, the Apple Word Handler uses <CONTROL> T (search Til) to initiate the search for a particular word or phrase, while Radio Shack SuperScripsit uses <@> G (global search) to perform the same function. Thus, while learning one word processing system aids in the understanding of word processing functions in general, each new system must be studied separately to identify its particular control-key commands.

Classroom Uses of Word Processing

Word processing software is available in a wide range of sophistication and complexity. Very simple programs with few commands to learn can be obtained, especially for the less expensive microcomputers such as the Commodore VIC and some Atari models. Medium and high-level programs can be purchased for virtually every brand of microcomputer. Prices for microcomputer word processing packages range from a low of around $40 to a high of $500-$700.

Teachers are becoming increasingly aware of the educational applications of word processing. The electronic editing and storage capabilities of this software aid the teacher in producing many classroom-related documents, such as student handouts and worksheets; assessment instruments; behavioral and performance contracts; letters and other correspondence to parents, students, and school personnel; and periodic reports on student performance and classroom activities.

Perhaps the most exciting application for word processing in the classroom, though, is the alternative it offers to those students who

have difficulty with the mechanics of pencil and paper tasks. Learning disabled, cerebral-palsied, language-handicapped and other special needs students are often unable or unwilling to produce written language because of the effort required to write legibly, integrate changes in wording or spelling, and produce clean drafts of written documents. With pencil and paper or typewriter, an acceptable final product is often produced only after erasing, reworking, and recopying again and again. These obstacles to change are discouraging to the struggling student and detract greatly from the actual task at hand--that of expression of knowledge or creative writing.

Word processing eliminates many of the tedious tasks associated with written expression. Legibility is no longer a problem; changes in spelling or wording are easily made without retyping the entire document; and neat, tidy drafts of the document can be printed at any time. The ability to produce edited, readable copy becomes a source of pride for the student, switches the focus from the mechanics of writing to creative expression, and motivates the student to engage in composition and written expression.

Word-processing programs

The following table lists the features of 11 word-processing programs CU has evaluated. We chose these programs in order to illustrate the range of features and prices available. Few home-users would need all the features listed here; extensive programs provide even more features than are listed. We suggest you use the column at the right as a worksheet, listing the features available on each of the several word-processing programs likely to be available for the computer you choose. The programs are listed in order of increasing price. In the columns, ✓ means yes, — means no.

Excellent ● Very good ◓ Good ○ Fair ◒ Poor ●

	COLOR SCRIPSIT $40	TELEWRITER 64 $50	VIC TYPEWRITER $60	WORD HANDLER II $60	BANK STREET WRITER $70	ATARI WRITER $100	APPLE WRITER II $150	EASY WRITER 1.1 $175	SUPER SCRIPSIT $199	WORD PRO 3 PLUS $295	WORD-STAR 3.0 $495	COMPARISON PROGRAM
Screen editing												
CURSOR: BY LETTER	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
BY WORD	—	—	—	✓	—	—	—	✓	✓	—	✓	
BY LINE	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	
BY PARAGRAPH	—	—	—	✓	—	—	—	✓	✓	—	—	
TO END/BEGINNING	✓	✓	—	—	✓	✓	✓	✓	✓	✓	✓	
DELETE: BY LETTER	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
BY WORD	✓	—	—	✓	—	—	—	✓	—	✓	✓	
BY LINE	—	✓	✓	✓	—	✓	✓	✓	—	✓	✓	
INSERT: BY LETTER	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
OPEN LINE	✓	✓	✓	✓	—	—	—	✓	✓	✓	✓	
RECALL DELETION	—	—	—	—	✓	✓	✓	✓	—	—	—	
BLOCK: MOVE	✓	—	—	✓	✓	✓	✓	✓	✓	✓	✓	
COPY	●	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
DELETE	✓	✓	—	✓	✓	✓	✓	✓	✓	—	✓	
SAVE	—	✓	—	✓	✓	—	✓	✓	✓	✓	✓	
SEARCH: AND FIND	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
AND REPLACE-ASK	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
AND REPLACE NO ASK	✓	—	—	—	—	✓	✓	✓	✓	✓	✓	
Screen formatting while editing												
COLUMNS	32	51	22	66	38	38	40	80	64	40	80	
LOWER-CASE LETTERS	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
LINE NUMBERS	—	—	—	—	—	✓	✓	—	✓	✓	✓	
PAGE NUMBERS	—	—	—	✓	—	—	—	—	✓	—	✓	
WORD WRAP	✓	✓	—	✓	✓	✓	✓	✓	✓	—	✓	
TABS	✓	✓	✓	✓	—	✓	✓	✓	✓	✓	✓	
PAGE BREAKS	✓	—	—	✓	✓	—	—	—	✓	—	✓	
Print formatting												
LEFT MARGIN	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
RIGHT MARGIN	—	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
TOP MARGIN	—	✓	—	✓	✓	✓	✓	✓	—	✓	✓	
BOTTOM MARGIN	—	✓	—	✓	✓	✓	✓	✓	—	✓	✓	
HYPHENATION	✓	—	—	—	—	—	—	—	✓	✓	✓	
HEADINGS	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
FOOTINGS	✓	—	—	✓	—	✓	✓	✓	✓	✓	✓	
LINE SPACING	1,2,3	1-3 & up	1	1 2⅓	1-3	1-3⅓ & up	1-3 & up	1-3 & up	1-3⅓	1 2 3	1 9	
CENTERING	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
JUSTIFIED MARGINS	—	✓	—	✓	—	✓	✓	✓	✓	✓	✓	
PRINT PAUSE	—	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
DEFAULTS	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	
CHARACTER ENHANCEMENT	—	✓	✓	✓	—	✓	—	✓	✓	✓	✓	
Conveniences												
SAVE AND RE-EDIT	✓	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	
HELP MENUS	—	—	—	—	—	—	✓	✓	✓	—	✓	
REFERENCE CARD	—	✓	—	✓	—	✓	✓	✓	✓	✓	✓	
TUTORIAL	✓	✓	—	✓	✓	✓	—	✓	✓	✓	✓	
REFERENCE MANUAL	—	✓	✓	✓	—	✓	✓	✓	✓	—	✓	
Judgments												
EASE OF LEARNING	◓	○	◒	○	●	◓	◒	◓	◒	○	◒	
EASE OF USE	◓	◓	◒	○	○	◓	◒	◓	◓	◒	◓	
	◓	○	●	◓	◒	◓	○	◓	◓	◓	◓	

WORD HANDLER

INTRODUCTION

Word Handler is word-processing system designed to permit the user to produce, edit, and store text-related documents on a microcomputer. Word Handler consists of one system disk which may be removed from the disk drive and replaced with any formatted document disk once the program has been booted. Thus, Word Handler can be used on a single-drive microcomputer, although, of course, a dual-disk drive may be more convenient.

HARDWARE REQUIREMENTS

Word Handler is developed for the Apple-II minimum microcomputer (48K, single or dual disk drive.) As with any word-processing system, a printer for hard copies is highly desirable.

SOFTWARE REQUIREMENTS

Word Handler consists of one program disk which contains the system utilities. Documents are stored separately on any formatted disk prepared with the INIT (initialize) program provided by the Apple's operating system (Apple DOS).

OPERATION

(a) Insert the program disk in the disk drive (If your Apple has 2 drives, insert the program disk in drive 1 and a formatted document disk in drive 2.)
(b) Turn the machine and monitor on.
(c) Word Handler will automatically boot itself and present you with a screen display with some version of the following:

PRINTER SLOT (1-5):
PRINTER TYPE (0-B):

	SERIAL	PARALLEL
ASCII (NO BKSP)	0	1
ASCII (W/BKSP)	2	3
OUME/DIABLO/TEC	4	5
EPSON MX-80/100	6	7
EPSON MX-80/EMPH	8	9
IDS-460/560	A	B

Use 66-COLUMN COMPACT FORMAT (Y/N)?

You need only answer the last question regarding the 66-column compact FORMAT. Typing a **Y** gives you a full-line display on the screen. Typing an **N** means you anticipate lines longer than 66 characters, and the lines will be broken up on the screen into two or three parts to allow all the text to be visible.

(d) Once you type Y or N (usually Y), Word Handler will be in its "idle" state, ready for you to enter the name of an old or new document. At this point, you may remove the system disk and insert the document disk into the disk drive (not necessary if you have 2 disk drives.)

NOTE: DO NOT REMOVE OR INSERT DISKS UNLESS WORD HANDLER IS IN ITS IDLE STATE. (Screen displays the "Enter Name..." display.) You can always return to the idle state by typing CTRL / E.

NAMING A DOCUMENT

Document names may contain up to 30 characters. To avoid having to type in all 30 characters each time you go back to this document place a semicolon near the beginning of the title. The next time you enter this document, you will only have to type the letters that appear before the semicolon.

For example, the first time you create a document you may name it **TEST1; FIRST SEMESTER FINAL**. The next time you enter this document you need only type **TEST1** (RETURN). Be careful not to name two documents with the same letters preceeding the semicolon.

Also, do not use the words: INDEX, PRINT, USE DISK 2, USE DISK 1/2, ERASE, RENAME, BACKUP, OR FILL-IN as the names of document files.

INDEX:	displays a list of all documents on the primary disk drive
PRINT:	prints pages of a specified document. Asks for document name and page numbers.
USE DISK 2:	causes all commands and controls to use disk drive 2
USE DISK 1/2:	causes disk drive 1 to be "primary", and disk drive 2 to be "secondary"
ERASE:	erases a document from a disk
RENAME:	gives a new name to a document
BACKUP:	makes a backup copy of a document
FILL-IN:	makes a new document from a "form" document while accepting "fill-ins" for all data items of the form: <things like this>

PRODUCING, EDITING, AND STORING TEXT

Once a document name has been entered from the idle stage, the program is automatically in EDIT MODE. If the document is an old one you may begin editing now. If the document is a new one, you cannot begin typing until you enter INSERT MODE by typing CTRL / I.

INSERTING TEXT

CTRL / I is used to begin entering text on a new document and also to insert a character, word, line, paragraph, or page into an old document. In the latter case, you begin in edit mode, place the cursor on the character where the new text is to begin, and press CTRL / I. Everything after the cursor will be blanked out (just from the screen) and new text may be typed in. When typing is finished, press the right arrow key to return to edit mode. The blanked-out text will then reappear after the insertion just typed in.

To enter text for the first time on a new document, from edit mode press CTRL / I to enter insert mode and begin typing the document. Various control-key combinations alter the entry of text during INSERT. These combinations are listed below and summarized on the Control Key Summary at the end of this guide.

CTRL / K	acts as a "caplock" until it is pressed again.
←	backspaces (erases type as well.)
→	terminates INSERT and returns to EDIT MODE.
CTRL / Y	starts entry of underlined text (ends with CTRL / N.)
CTRL / B	starts entry of **BOLD** text (ends with CTRL / N.)
CTRL / S	starts entry of "superscript" text (ends with CTRL / N.)
CTRL / N	ends entry of underlined, bold, and superscript text.
CTRL / V	changes vertical line spacing (can only be used at the beginning of a paragraph. Ends with CTRL / V.)
CTRL / J	changes justification of paragraph (can only be used at the beginning of a paragraph. Ends with CTRL / J.)
CTRL / O	sets and clears tab stops.
ESC	moves cursor to next tab stop.
CTRL / L	inserts an extra half-line space between lines (usu. para's.)
CTRL / P	forces an end to a page and skips to top of next page.
CTRL / W	creates an unbreakable space ("Mr.CTRL/WJones" avoids the possibility "Mr." will appear on one line and "Jones" on the next.)

EDITING TEXT

Once a document has been created, it can be changed in many ways when Word Handler is in the EDIT MODE. The following control-key combinations enable these changes:

MOVING THE CURSOR

→	moves forward one character at a time.
←	moves backward one character at a time.
CTRL / W	moves cursor one word at a time (pressing → or ← first determines the direction.)
CTRL / L	moves cursor one line at a time (pressing → or ← first determines the direction.)
CTRL / P	moves cursor one page at a time (pressing → or ← first determines the direction.)

DELETING TEXT

CTRL / D	deletes one character (can be repeated.)
CTRL / D, CTRL / W	deletes one word (can be repeated.)
CTRL / D, CTRL / L	deletes one line (can be repeated.)
CTRL / D, CTRL / P	deletes one page (can be repeated.)
→ or CTRL / I	actually finalizes any of the above actions.
←	cancels above actions if pressed before → or CTRL / I.

CHANGING PAGE FORMATS

CTRL / F allows you to change the following default values:

```
                          PAPER WIDTH:   8.5
                         PAPER LENGTH:   11

                           LEFT MARGIN:  1.0
                          RIGHT MARGIN:  0.9
                            TOP MARGIN:  0.8
                         BOTTOM MARGIN:  1.0

                      PITCH (10 or 12):  10
                    LINES/INCH (6 or 8):  6

                        FIRST PAGE NUMBER:  1

                     HEADER DIST FROM TOP:  0.4
                     FOOTER DIS. FROM BOT:  0.4

          ODD PAGE HEADER ("#"=PG NUM):
          ODD PAGE FOOTER ("#"=PG NUM):
          -#-
         EVEN PAGE HEADER ("#"=PG NUM):
         EVEN PAGE FOOTER ("#"=PG NUM):
```

use RETURN to skip any item you do not wish to change. TYPE over any entries you do wish to change. You can only proceed forward. If you want to change something above your current position, you must proceed through the format page and then use CTRL / F again, correcting the error the next time.

STORING TEXT

CTRL / E Once you have finished inserting text into or editing a document, you will wish to store the document on a document disk for later use. To accomplish this, simple press CTRL / E (for "end of editing") from either the INSERT or EDIT MODE. The document will automatically be stored under the name you specified earlier, and the Word Handler will return to its idle state. You may now request a new document to be opened, or you may remove the disk(s) from the drive(s) and turn the machine and monitor off.

PRINTING TEXT

PRINT — When the Word Handler is in its idle state you may print part or all of any document by entering PRINT when the screen prompts for "ENTER NAME..." You will then be asked for the name of the document to be printed and for the page numbers. If you want all the pages printed and know, for instance that your document contains 6 pages you could enter 1-6, 1-10, 1-100 or 1-anything above 6. The machine will stop printing when it senses there are no more pages in the document. If you don't know how many pages are in your document, set the upper limit to some number you know has to be above the actual number of pages and the printer will print the whole document.

If you do not want the entire document printed, just enter the page numbers you want. Separate the page numbers by a comma if they are not consecutive. Example: 1-6,9,12-14.

SEARCHING FOR AND REPLACING TEXT

CTRL / T — searches through the document for a word or phrase you supply (pressing → or ← first determines the direction of the search. Pressing → after CTRL / T ends the search.)

CTRL / R — requests both a word or words to be replaced, and what to replace it with. The cursor stops at the first occurrence of the specified word or words. Press CTRL / R again if you wish this occurrence to be replaced. Press RETURN if you do not wish this occurrence to be replaced (cursor moves on to next occurrence where you must make a decision again.)

aborts the replace operation if pressed either while specifying the words, or while the searching for the occurrences is happening.

CHANGING LINE SPACING

CTRL / V — cycles the vertical spacing between single, one-and-a-half, double, and two-and-a-half for the current paragraph only (can only be used when cursor is at the beginning of a paragraph.)

JUSTIFYING TEXT

CTRL / J — justifies the text of the current paragraph only; i.e., spreads text evenly over the length of the line. (can only be used when cursor is at the beginning of a paragraph.)

CENTERING TEXT

CTRL / X — centers the line the cursor is on. Typing CTRL / D-W- → uncenters text.

COPYING AND MOVING TEXT

CTRL / C — copies text you specify to be inserted into another position in the document.

CTRL / C, CTRL / W — copies a word at a time (can be repeated.)

CTRL / C, CTRL / L — copies a line at a time (can be repeated.)

CTRL / C, CTRL / P — copies a page at a time (can be repeated.)

→ or CTRL / I — actually finalizes any of the above actions.

← — cancels above actions if pressed before → or CTRL / I.

[any of the above actions causes the designated text to be copied into memory where it awaits your command to be reinserted elsewhere. NOTE: the designated text is not deleted from its original position.]

CTRL / I, CTRL / C — To insert the copied text at another location, move the cursor to the desired location and type CTRL / I-C.

CTRL / C, — copies a block of text to be inserted elsewhere in the text and deletes the block from the original position. ← cancels the copy and delete. → completes the copy and delete. To insert the copied text at another location, move the cursor to the desired location and type CTRL / I-C.

MERGING TEXT FROM OTHER DOCUMENTS

CTRL / G — gets text from another document and inserts it into a currently open document. Move the cursor to the desired location in the current document, press CTRL / G, supply the name of the second document and the line numbersof the text to be copied, and press RETURN. EX: FIRST1, 25, 29, 55-73, 16-18.
NOTE: the text is not deleted from the original document.

WORD HANDLER

CONTROL KEY SUMMARY

Key Strokes	EDIT MODE	INSERT MODE
CTRL / E	ends edit mode and returns program to idle state	
CTRL / I	switches out of edit mode into insert mode	
→ (Right arrow key)	moves cursor forward through text	ends insert mode and returns to edit mode
← (Left arrow key)	moves cursor backward through text	moves cursor backward through text (erases each character it passes over)
CTRL / K	turns caps lock on and off character or paragraph the cursor is on	turns caps lock on and off for each character subsequently typed until CTRL / K is pressed again
CTRL / V	changes vertical line spacing of paragraph in which cursor is placed	changes vertical line spacing of all text which follows until CTRL / V is pressed again
CTRL / L	moves cursor line by line (pressing the right or left arrow key first determines the direction in which the cursor will move.)	inserts an extra half-line space between lines (usually used between single-spaced paragraphs.)
CTRL / W	moves the cursor word by word (pressing the right or left arrow key first determines the direction in which the cursor will move.)	creates an unbreakable space. i.e., insures that two words will appear on the same line.)
CTRL / P	moves cursor page by page (pressing the right or left arrow key first determines the direction in which the cursor will move.)	forces a page break in the typed text

CONTROL KEY SUMMARY (Cont.)

Key Strokes	EDIT MODE	INSERT MODE
CTRL / D	enters delete mode. Using CTRL / W, L, & P now will mark characters after the cursor to be deleted. ← replaces characters marked for deletion. → completes the deletion and returns to the edit mode.	
CTRL / C	copies a block of text to be inserted elsewhere in the text. Using CTRL / W, L, & P now will mark characters after the cursor to be copied. ← cancels the copy function. → completes the copy and returns to the edit mode. To place the copied text elsewhere in the text, move the cursor to the desired location and type CTRL / I-C. NOTE: copied text has not been deleted from its original position.	
CTRL / C-D	copies a block of text to be inserted elsewhere in the text and deletes the block from the original position. Follow the instructions above to complete the copy and reinsertion.	
CTRL / T	TIL: searches through document until the first occurrence of a word or phrase you supply. (Pressing the right or left arrow key first determins the direction in which the cursor will move.) →, after CTRL / T, cancels the search.	
CTRL / R	replaces an unwanted word or phrase with a desired one. Once unwanted word is found, type CTRL / R again and the replacement is completed. Pressing RETURN, →, cancels the replacement.	
CTRL / J	justifies the paragraph in which the cursor is placed (cursor must be at paragraph's beginning.)	justifies anything typed from that point until CTRL / J is typed again

Key Strokes	EDIT MODE	INSERT MODE
CTRL / Y	underlines subsequent text until CTRL / N is typed	underlines subsequent text until CTRL / N is typed
CTRL / B		produces **bold** characters. Type CTRL / N to end bold.
CTRL / S		produces superior text (superscripts). Type CTRL / N to end superior.
CTRL / N	NORMAL: cancels underline	NORMAL: cancels underline, bold, and superior.
CTRL / Q		sets and clears tab stops
ESC key		moves cursor to next tab stop
CTRL / X	centers a word, phrase, or line (Typing CTRL / D-W- → uncenters text)	
CTRL / F	FORMAT: allows user to set the format for the printed page (margins, type pitch, headers, footers, etc.) NOTE: any changes made in format will affect the entire document.	
CTRL / G	gets text from a second document on the disk to insert into a currently open document.	

MICROCOMPUTERS IN SPECIAL EDUCATION

MODULE 4: Evaluating Educational Software

The majority of American schools now have access to microcomputers and are rapidly integrating computer-assisted instruction into their curricula. Literally thousands of educational software packages are available for this purpose. Unfortunately, much of the available software is unsuitable for classroom use either because of the designers' lack of knowledge regarding the educational process or because of their failure to anticipate the needs of individual teachers and students. Thus, it is the teacher's role to carefully evaluate instructional software with regard to certain universal requirements and to the specific needs of their students. The need for careful evaluation is particularly critical when the software is being considered for use in a special education classroom.

Objectives

At the end of this module, participants will be able to:

1. Identify and discuss recent general findings of research on computer-based instruction.

2. Identify the major potential benefits of computer-based instruction with special needs students.

3. Identify the four major critical features of instructional software that maximize instructional effects with special needs students.

4. Use specified criteria to examine and evaluate commercial instructional software for use with special needs students.

EVALUATING EDUCATIONAL SOFTWARE

One can scarcely pick up a newspaper or magazine today without encountering an article related to the "microcomputer revolution" sweeping the country. The move to integrate computer-assisted instruction into America's schools is well under way and gaining momentum daily. Recent reports indicate that over 50% of American schools already have access to microcomputers and that most teachers believe that microcomputer education is important for their students.

No one, of course, can predict the outcome of this movement with certainty. It is reasonable to predict, however, that the current microcomputer mania is more than just another educational fad. Unlike the passive learning provided by filmstrips, movies, overhead projectors, and television, microcomputers can individualize instruction, engage learners in an interactive instructional process, and instantly reinforce correct responses. These attributes make microcomputers ideal for teaching learning and behaviorally disordered (LBD) children.

Despite their capabilities we should never lose sight of the fact that microcomputers are, after all, "just machines" and no better teachers than the instructional software programmers. As LBD children require intensive instruction to enable them to close the gap between themselves and their age-mates in regular classrooms, it is important that microcomputer instruction be highly effective. To meet this requirement, microcomputer instructional software programs will have to be guided by model practices and controlled research.

GENERAL FINDINGS OF CBI RESEARCH:

Much of the research on computer-based instruction (CBI) has been carried out with populations other than LBD children. Results of this research may, however, provide clues to what we may expect when computers are used in the instruction of LBD students. General findings of CBI research, to date, are summarized below:

- Computer-based instruction combined with traditional instruction may be more effective than computer-based instruction or traditional instruction alone.
- The effects of computer instruction may be most dramatic with primary and elementary school children.
- Computer instruction may be most effective with children with low academic skills.
- Computer instruction with a variety of software programs has produced modest but positive gains in student achievement at higher instructional levels.
- Computer instruction may take less time to produce gains in students achievement than traditional forms of instruction.

- Computer instruction programs free teachers for providing additional instruction.

- Students develop and maintain a positive attitude toward computer instruction.

Because of the newness of microcomputer technology, however, we have little concrete information on which to draw to design effective microcomputer instruction for LBD children. Until recently, methods for teaching LBD children were based more upon speculation than fact; practitioners had to rely upon information drawn from case studies rather than from empirical research. Fortunately, recent results from both large-and small-scale studies have provided a sound technology for teaching LBD students. Translating these findings into effective microcomputer application, however, will require information drawn from further controlled research.

INSTRUCTIONAL POTENTIAL OF CBI:

The microcomputer holds considerable potential as an instructional tool. Among the instructional applications of computers are: 1) Drill and practice; 2) Tutorial; 3) Simulations; 4) Problem solving; 5) Instructional planning; 6) Computer managed instruction; and 7) Administration and record-keeping. Some of the salient instructional advantages offered by microcomputers as identified by Hanley (1983) are:

- an ability to structure to meet predetermined objectives;

- a heightened efficacy of "drill and practice" exercises with some students;

- the escaping of the long history of failure with workbooks;

- a nonthreatening instructional interaction;

- a failure-free mastery of new skills;

- the engaging and attention-keeping ability of microcomputers;

- the reinforcement and support for prior learning; and

- the savings in time spent to reach instructional objectives.

While studies have not clearly documented that computer-based instruction is solely responsible for increasing student achievement, they appear to hold considerable potential as instructional alternatives with LBD children who may not fare well with teachers or who display "learned helplessness" (Henker, Whalen, & Henshaw, 1980). These machines also offer automatic data collection and record-keeping options heretofore unavailable to teachers. Additionally, their potential for monitoring student progress on IEP's and for providing a

variety of labor-saving functions for bus/ teachers has not been fully exploited.

SOFTWARE SELECTION

Instructional programs for LBD students must be designed so that they can be adapted to the needs of LBD students and provide potent instruction. Due to the shortage of good educational software, LBD teachers face a major problem in integrating microcomputer technology into their educational programs (Hofmeister, 1982). Inadequate software may compound the difficulties faced by LBD students. If software programs fail to function properly, LBD students may refuse to use the programs and develop negative reactions to computer instruction in general (Rieth & Polsgrove, 1983). On the other hand, if software programs fail to provide adequate instruction or if they are used improperly, LBD students may not make sufficient academic gains (Rieth. Polsgrove, & Semmel, 1981; Rieth & Polsgrove, 1983).

Several factors explain the scarcity of appropriate software for use with LBD students. First, few teachers have the programming expertise to develop quality software and most commercial software developers are not professional educators. Also, in their rush to market educational software, developers often neglect to sufficiently test and "debug" their programs and rarely evaluate the educational effectiveness of the program. Finally, while some commercial software programs contain appropriate content, few incorporate the instructional and motivational features required to provide adequate remedial instruction for these children.

In selecting software, LBD teachers must initially concern themselves with what type of software they need: whether drill and practice. tutorial. simulation, problem-solving, or educational games. In view of the lack of available information concerning the effectiveness of various types of software with exceptional children, this is a difficult decision. As a first consideration it is recommended that the content of software be matched to the specific objectives identified for each student. Also in deference to the research related to the effectiveness of various instructional approaches and to the limitations of the currently available software, a conservative approach--the use of drill and practice and tutorial programs--for building academic skills may be most effective with this population.

This recommendation does not reflect negatively upon simulation and problem-solving approaches to instruction; these may be indicated with bright or gifted LBD children when enrichment activities are indicated. Mindful of the shortcomings of process and ability training with LBD students however. and to minimize confusion and motivational problems, it is not recommended that advanced, highly abstract, complicated. or operationally difficult software programs be used with these students.

Another choice facing LBD teachers involves when and how to use computer games. High interest educational games that required students to use skills learned in the academic program appear to be most beneficial. However, LBD teachers should use these cautiously. For example, a popular computer game is available that requires increasingly higher rates of correct answers to arithmetic problems for students to advance to higher skill levels. These

levels are displayed on the video screen. The program appears ideally suited for students who have appropriate skills who have no history of academic failure, and who are challenged by tasks of skill. However, because the screen format makes failure obvious, some LBD students might react to it with aggressive and/or withdrawal behavior for slower performing LBD students, this can lead to behavior management and motivational problems. Another available educational game teaches phonics using an "Adventure" format requiring students to move through a maze and place the video cursor in a precise position to get access to another room and another set of academic tasks. Not only is this program frustrating to operate, it wastes valuable instructional time. In considering a software program for use with LBD students you should be sensitive to any idiosyncracies, (1) that make it difficult to operate, (2) result in high amounts of lost instructional time, or (3) may precipitate behavioral problems.

There are also available a number of excellent games of skill and strategy (e.g., chess, "Othello") that may benefit LBD children's concentration and planning. While these skills are important, recent research suggests that they are best developed in the context of an academic instructional program (Meichenbaum & Arsonow, 1979). Because children have considerable access to arcade games and the content and skills developed in these are of questionable value, their use in an educational program may not be appropriate. In light of the ongoing discussion LBD teachers should carefully consider the use of all software programs. Because of the importance of time in instruction, LBD teachers should make access to computer games contingent upon meeting daily academic goals. Several excellent books have described how to establish and operate classroom contingency programs and LBD teachers should be well versed in these procedures (See Homme, et al. 1969; Kerr & Nelson, 1983; Hewett & Taylor, 1980).

Some writers have indicated that LBD children, who are making appropriate progress in the basic curriculum and express and interest, should be taught computer programming skills. If a teacher is qualified, and can locate appropriate training materials, teaching students to program might stimulate their academic motivation and classroom cooperation (Hannaford & Tabor, 1982). An alternative involves giving students canned programs (e.g. graphics, games) which they copy and debug as a learning and reinforcing activity.

Another potential computer activity for LBD students involves teaching them to write with a word processing program. Teachers could select one of the simpler ones on the market (e.g., Radio Shack's "Scripsit" program) and allow students to complete their writing assignments in this format. Although the instructional effectiveness of this approach has not been studied, this activity might be an effective strategy for stimulating LBD students in the use of language and/or for building interest, motivation and confidence in writing. Word-processing programs also require students to learn typing skills.

What are some of the critical features of instructional software that maximize instructional effects and minimize motivational problems with LBD children? How can LBD teachers use software to the best advantage? The newness of this technology makes these questions difficult to answer. However, based upon research related to academic learning time (ALT), direct instruction, applied behavior analysis, and studies on microcomputer applications in LBD classrooms some tentative recommendations can be offered:

1. Software programs for LBD students should include suitable supporting documents and materials.

Because microcomputer software programs are, after all, human products, their effectiveness hinges on the skill of their developers and the sophistication of their users. Hofmeister (1983) suggests that, at this stage in the development of the technology, teachers should regard software programs as supplementary instructional materials rather than as autoinstructional devices. As with using traditional instructional materials, then, teachers must understand the goals and content of an educational software program, identify students' instructional problems, and provide instruction at critical times in order to insure their effectiveness.

Unfortunately, few educational software programs include comprehensive teacher's guides similar to those that accompany printed instructional programs. To be maximally useful, these guides should describe the rationale of the program, list concepts covered, specify behavioral objectives, describe the procedures and features of the program, describe the branching features of the program, outline the scope and sequence of instructional activities, and list additional instructional activities. This information is crucial for LBD teachers to coordinate educational software with their larger instructional programs and with their students' individual educational plans.

2. Software programs for LBD children should provide direct instruction in academic skills.

The advent of microcomputer technology offers a renewed opportunity to change special education practices through the development of dynamic new instructional approaches. Because research supports a direct instructional approach to teaching LBD students, educational software for this group should attend to variables such as specifying and ordering objectives, placing students accurately in an instructional program, continuously monitoring student performance, adjusting instructional procedures to students' needs (i.e., "branching" students to higher or lower levels of instruction when required), providing frequent corrective feedback and reinforcement, maximizing academic learning time, and keeping student performance records for instructional decision-making (Rieth, Polsgrove, & Semmel, 1981). As minimal requirements, instructional software programs for LBD students should include the following features:

- clearly specified behavioral objectives and corresponding instructional tasks that follow an orderly sequence from least to most difficult.
- the introduction of potentially confusing concepts separately and the teaching of strategies after students have mastered basic skills (Silbert, Carnine, & Stein, 1981).
- a procedure for assessing students' entering skills and placing them at a starting point in the task sequence commensurate with their level of mastery of the material.

- continuous monitoring of a student's performance and altering the instructional content and methods based on this information.

- subroutines that analyze student errors and provide corrective instruction at critical points in the tutoring.

- functions that maximize students academic learning time: clear instructions, help keys, escape keys, and freedom from major idiosyncracies that require the teacher's aid.

3. Software should provide appropriate feedback and keep records of student performance within a program.

A number of studies have shown that informing students about the correctness of their responses (feedback)and graphing their academic scores positively affects their motivation (Barringer & Gholson, 1979; Polsgrove & Nelson, 1982). Microcomputers not only can provide clear, consistent, properly paced instruction and immediate feedback, but they can record the amount of time and accuracy of each student response. They can also quickly tabulate student performance scores and reproduce these as graphs or cumulative tables, products that LBD teachers can use for motivating students, for making instructional decisions, and for documenting students' progress on their IEP's. Educational software appropriate for LBD students should therefore provide the following functions:

- appropriate feedback concerning correct and incorrect responses.

- a graph or table summarizing each student's performance available in video and hard copy format.

4. Software should be under the teacher's control.

Above all, software for LBD students must be adaptable. If LBD teachers are unable to adapt it to the needs to individual students, they may abandon use of the programs altogether. Therefore, the software program should allow teachers to alter critical features in order to adapt it to meet the needs of their students. Specifically, teachers should be able to:

- change the presence or absence, frequency, and immediacy of the feedback functions of the program.

- change the amount and type of program prompts.

- change the criteria for success and for branching.

- determine the difficulty level of the content of the program.

- alter the remedial functions of the program (e.g., amount of review and practice given, time spent in remedial loops, and type of remediation.

Because idiosyncracies--"glitches"--appear in even good software programs, and because software evaluations based upon use for one group of students may be irrelevant for another, manufacturers' or user groups' software evaluations may be inaccurate concerning the suitability of a program for a group of LBD students. LBD teachers should, therefore, actually run a software program from start to finish and evaluate for themselves its appropriateness for use in their classrooms. The checklist which follows is intended to aid LBD teachers in evaluating software.

SOFTWARE EVALUATION FORM†

Directions: This form is intended to aid your decision-making in selecting appropriate software for use with LBD students. To use it, simply place your ratings and comments in the boxes to the right of each item. Use the following scale to arrive at a total weighted score for each software program. Note that four programs of similar type can be rated and compared on this form.

- 2 = Omission of or problems with this aspect of the program may have detrimental effects on students' learning and/or behavior
- 1 = Omission of or problems with this aspect of the program presents real difficulties in using it.
+ 1 = Program meets minimal acceptable standards on this aspect.
+ 2 = Program exceeds minimal acceptable standards on this aspect.

	Program Evaluated	Developer	Type of Program (e.g., CMI, CAI, Tutorial, Game)
1.			
2.			
3.			
4.			

CONTENT FEATURES

		1	2	3	4
1.	The teacher's manual provides enough information to use the program effectively.				
2.	The teacher's manual suggests appropriate supplemental materials and activities.				
3.	Students can follow the directions without aid.				
4.	The reading level of the video text is appropriate for students.				
5.	The content matches stated program objectives.				

† The author is indebted to F. Tabor for ideas on some items and general format.

SOFTWARE EVALUATION FORM

	1	2	3	4
6. The content corresponds to students' IEP objectives.				
7. The content is age-appropriate for students' social and intellectual developmental level.				
8. The program teaches important skills/information.				
9. The content is culturally appropriate for students.				
10. The content is free of cultural biases and stereotypes.				
11. The program holds students' attention.				
12. The content uses correct language and grammar.				
13. The content contains examples to clarify the material to be learned.				
14. The content adequately covers the topic to be learned.				
15. The information presented is accurate.				

INSTRUCTIONAL FEATURES

1. Long and short range program objectives are stated in observable and measurable terms.				
2. The instructional objectives are appropriately sequenced.				

SOFTWARE EVALUATION FORM

		1	2	3	4
3.	Prerequisite skills are clearly specified.				
4.	The program assesses student beginning skill-levels on various tasks.				
5.	The program places students in the program according to their assessed skill.				
6.	The instructional material in the software follows an appropriate sequence.				
7.	The program presents concepts that are likely to be confused, separately.				
8.	The program attempts to teach strategies.				
9.	Strategies are taught after students have mastered the prerequisite skills.				
10.	The program monitors students' responses and branches them to appropriate instructional levels.				
11.	The program provides specific tutoring or instructions when students respond incorrectly.				
12.	The program appropriately prompts students who are having difficulty.				

TECHNICAL ADEQUACY

1.	The program operates smoothly even when students make unexpected or "trick" responses.				
2.	The program appropriately limits the time that students spend on it.				

SOFTWARE EVALUATION FORM

		1	2	3	4
3.	The program safeguards confidential information on each student.				
4.	The program has a foolproof "log-in" procedure that keeps individual students' data separate.				
5.	The program is free from idiosyncrases that would frustrate students.				
6.	Program operations are reasonably fast.				
7.	Program menus are easy to use.				
8.	The program has easily accessible "help" menus.				
9.	The program records lapses in student responding (e.g., discontinuance, long latencies).				
10.	The program has an "escape" key that allows students to leave it when necessary.				
11.	The program allows teachers to change critical instructional features (e.g., rate of feedback, branching criteria).				
12.	A teacher can easily access student records.				
13.	Student records are easy to read and interpret.				
14.	Hard copy printout of student records is available.				

SOFTWARE EVALUATION FORM

		1	2	3	4
15.	The program provides appropriate feedback for correct responses.				
16.	The program provides appropriate feedback for incorrect responses.				
17.	Students are given immediate feedback.				
18.	The program provides accurate records of student performance. (e.g., bar graph, line graph, tables)				
19.	The graphics, color, sound, and illustrations used enhance the instruction.				
20.	The program presents concepts and feedback at a reasonable rate (i.e., is adequately paced).				
21.	The amount of time spent in the program is instructionally beneficial.				

Total Score: _____ _____ _____ _____

NOTES

GENERAL FINDINGS OF CBI RESEARCH

1. CBI COMBINED WITH TI MAY BE MORE EFFECTIVE THAN CBI OR TI ALONE.

2. THE EFFECTS OF CAI MAY BE MOST DRAMATIC WITH PRIMARY AND ELEMENTARY SCHOOL CHILDREN.

3. CBI MAY BE MOST EFFECTIVE WITH CHILDREN WITH LOW ACADEMIC SKILLS.

4. COMPUTER INSTRUCTION WITH A VARIETY OF SOFTWARE PROGRAMS HAS PRODUCED MODEST BUT POSITIVE GAINS IN STUDENT ACHIEVEMENT AT HIGHER INSTRUCTIONAL LEVELS.

5. COMPUTER INSTRUCTION PROGRAMS MAY TAKE LESS TIME TO PRODUCE GAINS IN STUDENT ACHIEVEMENT THAN TRADITIONAL FORMS OF INSTRUCTION.

6. COMPUTER INSTRUCTION PROGRAMS FREE TEACHERS FOR PROVIDING ADDITIONAL INSTRUCTION.

7. STUDENTS DEVELOP AND MAINTAIN A POSITIVE ATTITUDE TOWARD COMPUTER INSTRUCTION.

INSTRUCTIONAL POTENTIAL OF CBI

- Ability to structure to meet predetermined objectives
- Heightened efficacy of "drill and practice" exercies with some students
- Escape from the long history of failure with workbooks
- Nonthreatening instructional interaction
- Failure-free mastery of new skills
- Engaging and attention-keeping ability of microcomputers
- Reinforcement and support for prior learning
- Savings in time spent to reach instructional objectives

CRITICAL FEATURES OF INSTRUCTIONAL SOFTWARE FOR LBD STUDENTS

1. PROVISION OF SUITABLE SUPPORTING DOCUMENTS AND MATERIALS

2. DIRECT INSTRUCTION IN ACADEMIC SKILLS
   - Clearly specified and sequenced behavioral objectives and instructional tasks
   - Introduction of confusing concepts and strategies after mastery of basic skills
   - Assessment of entry skills and appropriate placement
   - Continuous monitoring of student performance
   - Error analysis and corrective instruction
   - Functions maximizing students' academic learning time

3. PROVISION OF FEEDBACK AND RECORD-KEEPING FEATURES

4. PROVISION FOR TEACHER CONTROL OF INSTRUCTION
   - Ability to change feedback functions
   - Ability to change program prompts
   - Ability to change criteria for success and branching
   - Ability to determine content difficulty levels
   - Ability to alter remedial functions

REFERENCES

Barringer, C., & Gholson, B. (1979). Effects of type and combination of feedback upon conceptual learning by children: Implications for research in academic learning. Review of Educational Research, 49(3), 459-478.

Hanley, K. S. (1983). The best computer books for kids. Classroom Computer Learning, 4(4), 32-34.

Hanneford, A. E., & Taber, F. M. (1982). Microcomputer software for the handicaped: Development and evaluation. Exceptional Children, 49, 137-144.

Henker, B., Whalen, C., & Hinshaw, S. P. (1980). The attributional contexts of cognitive intervention strategies. Exceptional Education Quarterly, 1, 17-30.

Hewett, F. M., & Taylor, F. D. (1980). The emotionally disturbed child in the classroom (2nd ed.). Boston: A'lyn & Bacon.

Hofmeister, A. (1982). Microcomputers in perspective. Exceptional Children, 49(2), 115-121.

Hofmeister, A. M. (1983). Microcomputer applications in the classroom. New York: Holt, Rinehart & Winston.

Homme, L. E., et al. (1969). How to use contingency contracting in the classroom. Champaign, Illinois: Research Press.

Kerr, M. M., & Nelson, C. M. (1983). Strategies for managing behavior problems in the classroom. Columbus, Ohio: Charles E. Merrill.

Meichenbaum, D., & Asarnow, J. (1979). Cognitive-behavior modification and metacognitive development: Implications for the classroom. In P. Kendall and S. Hollon (Eds.), Cognitive behavioral interventions: Theory, research and procedures. New York: Academic Press.

Polsgrove, L. (In Press). Microcomputer applications with LBD children. In E. Blackhurst (Ed.), Microcomputers in special education. Boston: Little, Brown & Co..

Polsgrove, L., & Nelson, C. M. (1982). Curriculum interventions within the behavioral model. In R. L. McDowell, G. W. Adamson, & F. H. Wood (Eds.), Teaching emotionally disturbed children (pp. 169-205). Boston: Little, Brown & Co..

Rieth, H. J., & Polsgrove, L. (1983). Evaluating and providing feedback on the effectiveness of instruction for handicapped children integrated in inner-city schools: Final report. Bloomington, Indiana: Center for Innovation in Teaching the Handicapped, Indiana University.

Rieth, H. J., Polsgrove, L., & Semmel, M. I. (1981). Instruction in the regular classroom: Variables that make a difference. Exceptional Education Quarterly, 2, 61-82.

Silbert, J., Carnine, D., & Stein (1981).

Tabor, F. M. (1983). Microcomputers in special education: Selection and decision-making process. Reston, Virginia: The Council for Exceptional Children.

MICROCOMPUTERS IN SPECIAL EDUCATION

Module 5 : Introduction To Authoring Systems

Authoring languages/systems are a popular tool for the development of educational software. Originally designed to be used by persons without extensive programming experience, authoring languages/systems are specifically oriented toward educational applications. This module will define authoring systems and authoring languages and describe their characteristics. The process of software development will then be discussed. Finally, the Apple II SuperPILOT authoring system will be discussed and reviewed in detail.

OBJECTIVES:

At the end of this module, participants will be able to:

1. List the advantages of using authoring systems in the special education classroom.

2. Name the major steps in the lesson development process.

3. Describe considerations in the development of instructional software.

4. Describe the overall organization of the SuperPILOT authoring system, including the names and functions of 4 editors and 2 operation modes.

5. Create and edit files within the SuperPILOT Lesson Text Editor.

6. Create and edit graphics within the SuperPILOT Graphics Editor.

7. Develop a SuperPILOT program for use in the special education classroom that utilizes at least 12 different SuperPILOT instructions.

INTRODUCTION TO AUTHORING LANGUAGES/SYSTEMS

Authoring languages/systems are a popular tool for the development of educational software. Originally designed to be used by persons without extensive programming experience, authoring languages/systems are specifically oriented toward educational applications. Authoring languages consist of a series of commands or instructions, which typically enable a program to:

Present text and questions on a computer screen.

Accept student responses entered through a keyboard or other peripheral device.

Analyze student responses.

Check for errors in program operation and program use.

Store responses and counter values.

Branch to other parts of the program, based on student response.

Provide feedback.

Maintain records of use and performance.

Interface with subroutines written in other computer programming languages.

Use graphics, animation, and/or sound effects.

Authoring systems include a built-in organizational structure for the use of instructions or commands. For example, a series of commands that perform similar functions, such as graphics creation, might be organized into a separate subsystem within an authoring system.

Authoring languages/systems offer several advantages over programming languages for the development of educational software. Typically, they are less complicated than programming languages, with fewer features and commands. They include predefined subroutines and functions that are relevant to instructional applications. In addition, authoring systems eliminate the necessity to learn the computer's operating system, as commands are packaged in such a manner that the user need interact only with the authoring system itself. These advantages make authoring languages/systems somewhat easier to learn than programming languages, thus reducing the cost and effort of developing instructional software.

However, there are some disadvantages to authoring languages/systems. As described by Merrill (1982):

> "authoring systems reduce cost and effort by reducing variety in the same way that cost and effort are reduced in fast food restaurants".

Since fewer commands are available to the user of authoring languages/systems, the resultant software may be less sophisticated than software developed with programming languages. Some authors believe that the limitations imposed by authoring systems encourage the development of stereotypical or mediocre lesson formats. Moreover, since a specific authoring language/system is restricted to use on one type of machine, authoring languages/systems are not as transportable as programming languages. Finally, authoring languages/systems cannot be mastered instantaneously. Although they may be learned more quickly than many programming languages, it still requires time and effort to learn an authoring language/system thoroughly.

SOFTWARE LESSON DEVELOPMENT

The development of instructional software can be described as a five-step process: 1) preparation, 2) writing the main lesson, 3) adding special effects, 4) evaluation, and 5) testing. These five steps are flow-charted in Figure 1, on the following page.

In the preparatory phase, the teacher conceptualizes lesson activities before sitting down at the computer and actually using the authoring language/system. A thorough and complete conceptualization of any software lesson will help prevent the development of programs that don't run as expected or don't meet the user's needs. First, the user of an authoring language/system must define the educational objectives he/she wishes a student to attain. The teacher should then consider how these objectives will be presented in a lesson and write an outline that describes lesson activities and the sequence of those activities. In the final step of the preparatory phase, the user needs to consider how activities will be grouped into different program sections.

In the second phase of lesson development, the teacher begins to work within the authoring language to translate lesson ideas into language instructions and routines. At this stage, mistakes will be easier to detect and correct if the program is given frequent test runs. Therefore, the teacher should run the program after each section is written, to ensure that the program is working as expected. It is also critical to back up the lesson disk after each section of the program is written and debugged, so that valuable work isn't inadvertently destroyed.

The third phase of the lesson development process consists of adding special effects (such as graphics, animation, and sound effects), placing these into the program, and test running the program. Once again, the lesson should be backed up to an alternate disk. The lesson is then ready for student testing, and the teacher may want to watch as students run the lesson to make sure that the program runs as planned and that students respond as anticipated. In the final phase, the teacher may wish to evaluate the lesson's appropriateness and effectiveness and make modifications where necessary.

In order to develop software programs that are both effective and enjoyable to use, the user of an authoring language/system must attend to instructional design considerations when developing instructional software. Typically, the commands available within an authoring language simplify the

task of incorporating characteristics of sound instructional design, such as branching and error checking. Moreover, capabilities of the microcomputer itself, such as the availability of color and sound effects, can be used to enhance the presentation of instructional material.

Five related considerations should be addressed in the design and development of computer courseware. These are summarized below, and described in more detail within the two articles included in this module.

1. Program sequence

   a. Does the program check for and rectify errors in student response that may prevent the program from running or cause the program to run in an unpredictable manner (e.g. the student types the name of a number rather than typing a numeral).

   b. Does the program branch students to different parts of the program, depending on their responses?

   c. Is the program variable, i.e. does it run a little differently each time it is used?

   d. Does the program provide a means for the student to exit upon request?

2. Cognitive characteristics of student users

   a. Does the program present one idea at a time on the screen?

   b. Does the program provide sufficient time for the student to process the information presented on the screen?

   c. Are more difficult concepts presented in more detail or depth?

   d. Is the vocabulary appropriate for the learners who will use the program?

   e. Does the program provide adequate repetition of concepts or activities?

   f. Is the program's length appropriate for the attention span of the learners with whom it will be used?

   g. Does the program use natural and consistent response formats (e.g. a "yes" answer is always indicated by pressing the "y" key).

   h. Does the program cue and prompt the student, where necessary, to encourage correct responding.

3. Feedback

   a. Is immediate feedback provided?

b. Is feedback provided intermittently, rather than after every correct answer?

c. Is feedback informative, i.e., will it help the learner to make a correct response in the future?

d. Is feedback nonthreatening?

e. Is feedback provided in an age-appropriate manner?

4. Data collection

a. Are records kept of student performance, including records of accuracy, rate, and latency of response, where appropriate?

b. Are records kept for later use by the teacher or student?

c. Is student performance data used to diagnose and prescribe future instruction?

5. Program appearance

a. Do graphics and visual features of the program support the concepts being discussed, rather than distracting from them?

b. Are colors and textual cues (e.g. boldface, double-sized letters) used to highlight important information and to assist the learner in making discriminations among concepts?

c. Are the visual features of the program age appropriate?

SuperPILOT is an authoring system designed to be used with an Apple II or Apple II Plus microcomputer with dual disk drive. The next section of this module will review selected aspects of SuperPILOT; including an overview of the SuperPILOT system, use of the lesson text editor and the graphics editor, and an overview of commonly-used SuperPILOT instructions and functions. A list and short descriptions of other authoring languages/systems is provided in the final section of this module.

DEVELOPING A SUPERPILOT LESSON

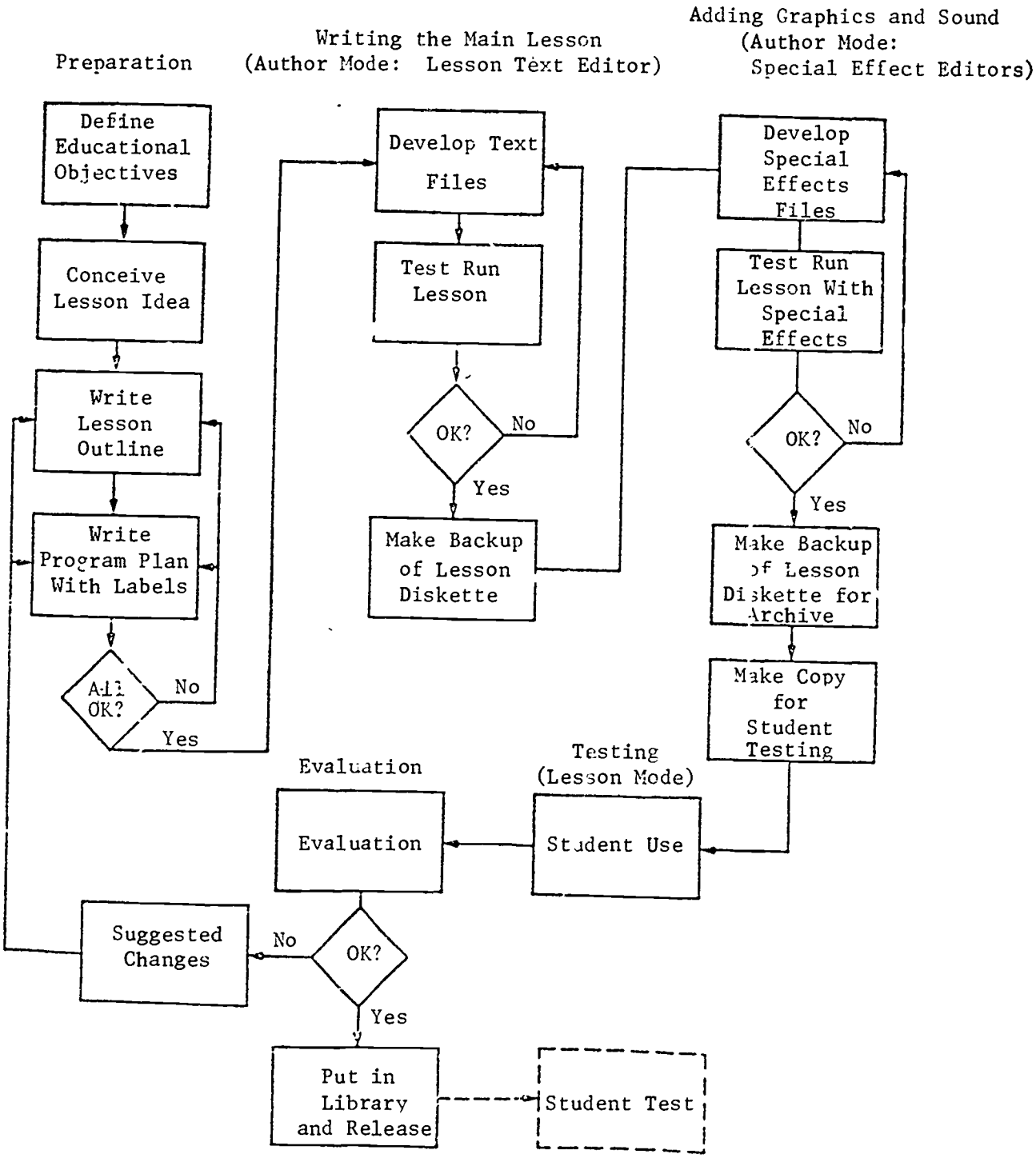


Figure 1-5: . Lesson Development Diagram

OVERVIEW OF SUPERPILOT

SuperPILOT is a descendent of the popular PILOT authoring language, which was developed in the 1970s specifically for instructional applications. SuperPILOT includes four editors that enable the user to create lessons that use graphics, sound, and different character-sets. The SuperPILOT language, which forms the backbone of the system, consists of 26 instructions, of which 9 are most commonly used. Although SuperPILOT typically accepts student responses that are typed in from a keyboard, lessons also may be modified to accept responses from game controllers, a light pen, or a touch screen.

SuperPILOT can be used in two modes: author mode, which permits you to create lessons, edit lessons, and interact with lessons from the student's point-of-view, and lesson mode, which is used by a student to run previously created SuperPILOT lessons. Author mode requires two disk drives - one to hold the author disk and one to hold a lesson disk. Lesson mode requires only one disk drive.

Authoring Mode

1. Place the Author disk in drive 1 and Lesson disk in drive 2.
2. Boot the system.
3. The Main Menu will appears, which offers 6 choices for the use of Author Mode:
   a. Lesson Text Editor to create a lesson by typing instructions in the SuperPILOT language.
   b. Character Set Editor to create a new set of designs or characters that can be used in a SuperPILOT lesson.
   c. Graphics Editor to draw a picture or graphics image that can be used in a SuperPILOT lesson. Graphics files may be stored in two ways: standard form, which redraws a picture, step-by-step, on the screen; and quickdraw from, which brings a finished picture to the screen.
   d. Sound Effects Editor to create short musical compositions or sound effects that can be used in a SuperPILOT lesson.
   e. Initialize a diskette to format and initialize a diskette for use as a lesson diskette.
   f. Diskette copy utility to copy a lesson diskette to an initialized diskette.

Lesson Mode:

Place a lesson disk in drive 1 and boot the system. Upon entering Lesson Mode, a student may be shown a menu of all the lessons on that disk and asked to choose a lesson. Or, he/she may be automatically placed into a lesson through the use of the Hello lesson.

Hello lesson: when the student boots the system with a lesson disk in Drive 1, SuperPILOT looks for a lesson on the diskette entitled 'Hello'. If found, this lesson is immediately run. An actual lesson can be created and stored as the Hello lesson, or the Hello lesson can run another lesson, or the Hello lesson can present to the student a restricted menu of lesson choices.

File lengths in SuperPILOT:

Each lesson diskette is formatted into 280 blocks. Of these, 144 are available for your use, the remaining 136 are occupied by system information. Upon entering any editor, SuperPILOT tells you how many blocks remain on your lesson diskette. The following presents guidelines for the allocation of disk space in each editor:

standard graphics files - 0 to 127 commands = 1 block
quick-draw graphics files - 0 to 127 commands = 17 blocks
character set files - 0 to 96 commands = 2 blocks
sound effects files - 0 to 100 notes = 1 block

Space allocation in lesson text files is more variable. Each lesson requires two blocks to hold system information. Space is then allocated two blocks at a time, and each 2-block allocation holds approximately 1,000 characters. Therefore, each lesson will occupy a minimum of four blocks.

Common Editor Features

All four editors share some features in common. First, they are all accessed from the Main SuperPILOT menu. Upon entering an editor, you will always be told the amount of remaining space on the lesson diskette and the names of the files presently residing on that editor. Twenty-four files can be created and stored under each editor. Options common to all editors are:

New to begin a new file. You must first assign a name to the file, which can be from 1 to 10 characters in length and cannot include spaces or special characters.
Edit to create a new file, view a file, or make changes in a file.
Run to run a lesson created in the lesson text editor. This option simulates the lesson mode, from the student's point-of-view, except error messages will be provided to help you debug your lessons.
Print to print a file. Files created by the graphics and character set editors can be printed only with an Apple Silentype Printer.
Delete to permanently delete files from a lesson diskette.
Quit to return to the Main SuperPILOT Menu.

Saving Files

Files are saved from inside each editor with the Q (quit) command. Files can be saved with changes, or you can quit without saving your changes. In addition, you can quit an editor and rename a file. Upon saving a file, SuperPILOT will inform you of that file's length. If a file is too long, you might not be able to save it without special modifications. Always have a spare initialized diskette on hand; if a file is too long, you might be able to save it on a new diskette.

SuperPILOT Lesson Text Editor

General Commands

Cursor set direction:
+ or > indicates forward direction
- or < indicates backward or reverse direction

Replace, Find, Page commands are affected by cursor direction, e.g., Find with a forward cursor direction searches from the present cursor position forward, to the end of the file.

Repeat factor:
A number typed immediately before a command designates the number of times that command will be repeated. This can be used with cursor moves and Find and Replace commands.

ESC key:
Saves changes made from Insert, Delete, and Exchange commands.

CTRL-C:
Cancels a command.

M (memory):
From the edit level, type M to find out how many characters you've used and how many characters are still left in useable memory. STOP inserting text when there are less than 250 remaining characters.

Q (quit):
Ends an edit session and prompts you to save and/or rename the lesson being edited.

Cursor moving commands

Right arrow key: forward or to the right
Left arrow key: backward or to the left
CRTL-O: up.
CTRL-L: down.
CTRL-I: tab (permanently set at every 8 spaces)
P (page): next page (23 lines per page).
RETURN key: next line.

To speed cursor movement, press REPT key while issuing above commands OR precede commands with a repeat factor.

J (jump): To jump to another section of the text, press J while at edit level, then:
B (beginning): to jump to first character in the file.
E (end): to jump to last character in the file.
L (label): to jump to a specified label (you will be prompted for the label name).

F (find): To find a target string, press F while at the edit level and specify the target string by enclosing it in delimiters (a set of identical punctuation marks). The following options may be used with Find:

Repeat factor: type the number of the repeat-factor before typing F.

Literal search: enables you to find a target string that is embedded in a larger string. Press the L key after pressing the F key.

Same string: enables you to find the same string you specified in a previous Find or Replace command. Type FS to find the same target string or FLS to make a literal search for the same target string.

Text-changing commands

I (insert): permits you to insert text, with the following options:

<= use the left arrow key to backspace and change characters added since you entered the Insert level.

ESC: to leave Insert level and save insertions.

CTRL-X: to delete the entire line you are inserting.

CTRL-C: to leave Insert level and cancel insertions.

D (delete): deletes text from a lesson by moving the cursor over the text, using the cursor moving commands.

ESC: to leave Delete level and accept deletions.

CTRL-C: to leave Delete level and cancel deletions.

X (eXchange): to simultaneously delete old text and insert new text on a one-to-one basis. Exchange commands are limited to a single line, to make changes on a subsequent line you must accept the exchange on one line and then move the cursor to the next line and re-enter the eXchange command.

C (copy): to copy large blocks of text into a file, using the following options:

B (buffer): copies your last insertion or deletion into the text, beginning at the present cursor position. This enables you to recover text you accidentally deleted, or to move sections of text by deleting them from one location and then copying them into another location. Blocks of text may also be copied without actually deleting the original text by:

a. enter the Delete level and delete the block of text you wish to copy

b. leave Delete by using CTRL-C, which cancels the deletion and thus retains the original text. However, CRTL-C causes the block of text to be copied into the copy buffer anyway.

c. Press CB to insert the text at the present cursor position.

F (file): copies an entire file into the lesson, at the present cursor position. Only lesson files may be copied.

CTRL-C to cancel the copy command.

R (replace): to substitute one sequence of characters for another. The string to be replaced is indicated first, followed by the substitute string. Delimiters must be placed at the beginning and end of each string, e.g. /100//one hundred/, /good//better/. Replace operates in a similar manner to the Find command, and uses the same options (repeat-factor, literal search, and same string).

Graphics Editor

Overview: The graphics cursor is the primary tool for the creation of images within the graphics editor. Cursor movement and form commands enable you to draw images on the screen that you later save as standard graphics files or quickdraw files. Although you see a visual image on the screen as you work, the graphics editor is actually creating a sequence of instructions, similar to those you create within the lesson text editor. Your graphic is stored as a file of instructions, rather than in the form of a visual image.

Getting Information: Two kinds of information are available in the graphics editor:

Help screen: The help screen appears each time you begin an editing session. It serves as a reminder of the commands available to you within the graphics editor. You may recall the help screen at any time by typing a question mark (?).

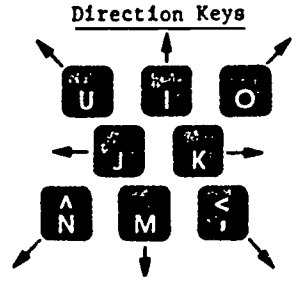
W (What?): Pressing the W key whenever your graphic is on the screen provides you with information about the current status of that graphic, including the last color to be selected, the present position of the cursor, the last command issued, and the cumulative number of commands you have issued thus far in the graphic.

Cursor Movement: SuperPILOT graphics are drawn by a rubber band cursor, which appears as a small blinking dot upon entering the editor. The cursor can be stretched in any direction and to any length within the boundaries of the screen. When the cursor is positioned where you wish, you can then draw a line along its length or form several shapes. Typically, the cursor is manipulated by the keyboard, but game-paddle controls may also be used.

Keyboard control: The keys U, I, O, J, K, N, M, and comma (,) move the cursor in specified directions as indicated below:

You type	Result
U	up, left
I	up
O	up, right
J	left
K	right
N	down,left
M	down
,	down, right

Direction Keys



The cursor stretches one dot at time, and the REPT key may be used to repeat a command (hold down the REPT key and appropriate letter key simultaneously).

> and < keys: > key moves the cursor in larger steps (7-8 dots at a time). Press the < key to return to regular cursor movement.

S (skip): establishes a new starting position for the cursor. After stretching the cursor to the desired position (through the use of appropriate letter keys), press the S key.

Creating Graphics Images: The editor does not add anything to your graphics file until you give one of the following commands:

D (draw): when you have positioned the rubber band cursor where you want a line to appear, press the D key. A line is drawn along the cursor in the color that you last specified.

F (frame): after drawing a line with the rubber band cursor, F draws a rectangle (or frame), using the rubber band cursor's position as the diagonal.

C (circle): after drawing a line with the rubber band cursor, C draws a circle or an oval, using the rubber band cursor's position as the diagonal.

B (box): like the Frame and Circle commands, the B command draws a solid box with the rubber band cursor's position as a diagonal.

A (area fill): the Area Fill command fills in the enclosed black area of any shape with the current drawing color. To use this command, the cursor must first rest within the area to be filled.

Graphics Colors:

Six different colors are available in the graphics editor, as follows:

0=Black1	3=White1	6=Blue
1=Green	4=Black2	7=White2
2=Violet	5=Orange	8=Reverse

Upon entering the graphics editor, the colors are automaticaly set to Black1 (0) for screen background and White1 (3) for drawing color. At any time, drawing colors may be changed by pressing the number of the desired color before issuing a Draw, Frame, Circle, Box, or Area Fill command. Screen background can be changed by stretching the rubber band cursor from one corner of the screen to the other, typing the number of the desired color, and issuing the Box command.

To make a line or other figure stand out across colored areas, you can use number 8 for Reverse. Reverse changes every color it passes over to the complement of that color, as follows:

Black1-->White1	White1-->Black1	Blue-->Orange
Green-->Violet	Black2-->White2	White2-->Black2
Violet-->Green	Orange-->Blue	

T (type text): Labels, titles, and other text can be placed on the screen with the T command. Place the cursor wherever you want text to begin, and then press the T key. The keyboard may now be used as a typewriter; however each line of text must be entered separately by typing a line and then the ESC key to accept that line.

Editing Graphics

Moving about in the file : the left arrow and right arrow keys permit you to move forward and backward, respectively, through the graphics file. ESC jumps to the end of the file, and W, when pressed, indicates the status of your graphic.

Making changes : Commands may be added into the file at any point. Additional options include:

E (erase): erases a command.

R (restore): puts a previously erased command back into the file.

V (verify): updates the graphic on the screen with the modifications you have indicated.

Relocate Origin: By default, the origin of all graphics is the center of the screen. However, the graphic can be relocated by using the rubber band cursor to indicate a new origin and then pressing the * key.

QuickDraw: To store a graphics image as a quickdraw file, give it a name that includes an exclamation point (e.g. DOODLES!, !DOODLES, DO!ODLES). Quickdraw files are edited in the same manner as standard files.

SuperPILOT Language Reference Sheet

Overview:

The Apple SuperPILOT programming language consists of twenty-six instructions or commands. Each command has two main parts: the instruction itself and an object. Most instructions begin with a single-letter instruction name, followed by a colon. It makes no difference to SuperPILOT whether instruction names are in upper or lower case, they will be executed in the same manner regardless.

In the lesson text editor, each line of text can be only 39 characters in length. SuperPILOT will automatically continue instructions on subsequent lines, up to 250 characters total, by "wrapping around" and starting each continuation line with another colon.

There are two ways to alter the execution of a particular instruction. A modifier changes how an instruction works. For example, a Type instruction with a hang modifier (e.g. th: 7 + 6 =) causes the cursor to "hang" at the end of the displayed text, rather than moving to the next line. A conditioner causes an instruction to be executed or skipped, often depending on the student's response.

A label is used to identify the beginning of a program section and can be referred to in subsequent instructions. A label must start with an asterisk (*) and may have a name of up to 35 characters, of which only the first six are meaningful to SuperPILOT (therefore, make sure the first six characters of all your labels are unique).

Basic commands:

T (type): indicates text to be typed on the screen.

T: with no object places a blank line on the screen.

TH: causes the cursor to remain at the end of the text, rather than automatically moving to the next line. (When you use TH, remember to type out a few blank spaces if you want to separate the question from the answer.)

A (accept): used to accept a student's response and store it, temporarily, in an "answer buffer". The contents of the answer buffer remain unchanged until another A: instruction is encountered.

AS: enables the computer to accept a single character response (e.g. T, F, Y, N, carriage return).

AX: accepts the student's response exactly as typed, with no response editing, temporarily overriding PR instructions.

M (match): specifies items to be matched to a student's response. Items in the object field of the match instruction are compared to the contents of the answer buffer. Controller conditions limit and/or expand acceptable matches as follows:

! - a match is made if any of the items in the instruction's object field appear in the student's response.

& - a match is made only if all items in the instruction's object field appear in the same order in the student's response.

% - a match is made if no characters other than a space appear in the student's response in the position corresponding to the %.

* - a match is made if any character appears in the student's response where the * appears in the instruction's object field.

MS - up to one character may differ in the student's response.

MJ - if the student's response does not match the object, jump to the next match instruction.

J (jump): causes the program to branch to a specified label in another part of the lesson. The object field of the Jump command indicates the destination of the jump. A jump can be conditional upon the following:

a match is found - JY
no match is found - JN
a specified number of incorrect answers - e.g. JY1, JN2.
the error flag is up - JE

You may also jump to an unlabeled destination as follows:

@A - the last accept instruction previously executed
@M - the next match instruction
@P - the next PR instruction

U (use): causes the program to branch to the label or to the unlabeled instruction specified in the object field. Unlike the Jump instruction, the Use instruction "remembers" where the branching occurred and returns to that spot when an End instruction is encountered. Typically, the subroutine referred to in a Use instruction begins with an identifying label and ends with an End instruction.

W (wait): temporarily stops the lesson for the number of seconds indicated in the object field of the instruction.

R (remark): leaves notes or comments in the program for that are not executed by the computer.

D (dimension): enables you to store variables for use throughout the program by setting aside storage locations in memory. Two types of variables may be stored:

numeric variables: indicated by # and a single letter or a single letter and integer (e.g. #a, #b1). Computations can be performed only on numeric variables. Single numeric variables need not be dimensioned.

string variables: indicated by a single letter or a single letter and integer, followed by a $ (e.g. t$, z1$).

In the dimension command, the variable must be named and the size of that variable specified in the object field. Size is indicated in parentheses, and refers to the maximum number of characters that variable will require (e.g. D: t$(60)).

The Accept command can be used to accept student input and store it in a numeric or string variable, as shown below. Note that a space must precede and follow variables when they are used in T instructions.

```
D: z$(10)
T: Please type your name.
A: $z$
T: Hello,  $z$ , how are you.

T: How old are you?
A: #a
T: So you are  #a years old?
```

C (compute): The compute instruction enables a fixed or computed number to be stored in a simple numeric variable or in an element of a numeric-variable array. For example, the command c:n=2+(3/x) calculates the current value of 2+(3/x) and stores this value in the simple numeric variable n. The compute instruction also may be used to store a fixed or computed string of characters in a string variable. For example, the command c:s$="Apple" stores the characters Apple in a string variable s$.

PR (problem): this instruction changes aspects of program operation starting at the point it is encountered in your program. PR options include:

Response editing - edits a student's responses by converting all characters to lower case (PR:L), converting all characters to uppercase (PR:U), or removing all spaces from responses (PR:S).

Timing responses - The command PR:Tn sets a maximum time of n seconds for response to be emitted.

Goto - The command PR:G permits you to use the Goto command as a program is running to jump to a specified label or unlabeled destination.

Graphics and special effects commands

GX (graphics execute): enables you to call a specific graphics file and display it within the lesson. The name of the file is indicated in the object field of the instruction.

TS (type specify): modifies the Type instruction to indicate how the text will appear on the screen. Options include:

TS:Vl,r,t,b (text viewport): enables you to set a window for the display of text. Default is the full screen (TS:v0,39,0,23).

TS:Sn (size of type): sets the size of text characters where n equals 1 (regular size) or 2 (double size).

TS:Tn (thickness of type): sets the thickness of type where n equals 1 (normal type) or 2 (boldface type).

TS:ESn (erase screen in graphics color): erases the entire screen in the graphics color specified by n. Options for n are:

0 (black1)	3 (white1)	6 (blue)	9 (single-dot on)
1 (green)	4 (black2)	7 (white2)	10 (single-dot off)
2 (violet)	5 (Orange)	8 (reverse)	

TS:Bn (background text color): sets the background color of the screen to the color indicated by n. Twenty-one colors are available, including 0-7 in the list above.

TS:Fn (foreground text color): sets the color of the foreground used to display text on the screen, where n indicates the desired color, from 0 to 7, as listed above.

Conditioners

A conditioner is an optional element of an instruction; it states a condition that must be true or the rest of the instruction will not be executed.

Y and N (yes and no) conditioners: These conditioners test the success of the last match instruction. The Y conditioner executes an instruction only if the previous match was successful and the N conditioner executes an instruction only if the previous match was unsuccessful.

Answer-count conditioner: The answer-count conditioner indicates now many times in a row the most recent Accept command has been encountered. This value (from 1 to 99) is stored in a system variable called %a. The answer-count conditioner permits an instruction to be executed only if the value stored in %a is equal to the number you specify. For example, TY1: will be executed only if the student's response matches and the value of the answer count conditioner is one (i.e. the student has attempted the problem once before obtaining a match).

E (error) conditioner: Whenever SuperPILOT has difficulty executing one of your instructions, it raises an error flag. (When you run a lesson in the Author mode, the error messages you see are triggered by instructions that have raised the error flag). When the error conditioner is attached to an instruction, that instruction is executed only if the error flag has been raised. This conditioner allows you to handle certain predictable errors in the operation of your program and/or in students' responses. For example, JE: will cause the program to jump only if the error flag is up.

Expression conditioners: Expression conditioners perform a calculation or make an assertion, causing an instruction to be executed or skipped depending on the result of the calculation or assertion. If the expression is true, the command is executed; a false result causes the command to be skipped. For example T(n=12): will be executed only if the variable n equals 12.

C (last expression) conditioner: The last-expression conditioner is a convenient shorthand notation for the instruction-modifying expression that was last evaluated in your program. For example, following the instruction T(n=12): you may write JC: to indicate "jump if the variable n equals 12" (since n=12 was the last instruction-modifying conditioner).

Functions:

SuperPILOT has a large number of special functions that perform specific operations. More commonly used functions include:

FIX(X): returns an integer in the range -32767 through 32767. found by truncating X.

RND(X): returns a random number in the range determined by the value of X.

SQR(X): returns the square root of X.

TIM(X): returns a number that is the time, in seconds, that elapsed while the last Accept instruction waited for the student to complete a response.

HELLO LESSON

When a student uses SuperPILOT in lesson mode, if a lesson called HELLO is on the diskette, that lesson will be run immediately. If not, a menu of available lessons will be presented to the student. A specific lesson may be created and named HELLO, or the HELLO lesson may consist of a one line statement that links to another file on the disk. For example, a HELLO lesson that contains the instruction L:Math will run the program entitled Math anytime the system is booted in lesson mode.

Lesson Text Editor

Use the following SuperPILOT program to practice using the Lesson Text Editor. First, enter the Lesson Text Editor:

Place the Author disk in drive 1 and the Lesson disk in drive 2.
Boot the system.
When the main menu appears on the screen, press L for Lesson Text Editor.

Within the Lesson Text Editor, enter the Insert mode by pressing I. Type in the text that is listed in the left hand column of the following program. (The right hand column provides a brief description of the program's operation). Then, practice editing your program, using the commands listed on the next two pages.

t:Studying superheroes can tell us a	(Display this text.)
:lot about our own culture.	(Continue text display.)
t:	(Insert blank line.)
t:Have you ever heard of Wonder Woman?	(Display this text.)
t:	(Insert blank line.)
a:	(Accept answers.)
m:yes!yeah!sure!yep!uhhuh	(Match to these answers.)
ty:Name one superhuman quality that she :exhibits	(If a Match is found, display this text.)
jy:part1	(If a Match is found, jump to the label named part1.)
t:Have you ever heard of a different :superhero?	(If no Match is found, display this text.)
a:	(Accept answers.)
m:yes!yeah!sure!yep!uhhuh	(Match to these answers.)
ty:Name a superhuman quality associated :with that hero	(If a Match is found, display this text.)
jy:part1	(If a Match is found, jump to the label named part1.)
t:I'll bet you don't watch much TV or :read many comic books! Since you've :never heard of a superhero that :someone else has made up, make up your :own! Then name a superhuman quality :he or she possesses.	(If no Match is found, display this text.)
*part1	(This is the label named part1.)
a:	(Accept answers.)
t:Would you like to have that :superhuman quality?	(Display this text.)
a:	(Accept answers.)

Graphics Editor

Follow this set of instructions to create a graphics "doodle" in the Graphics Editor. First, enter the Graphics Editor by:

Placing the Author disk in drive 1 and the Lesson disk in drive 2.
Booting the system.
When the main menu appears on the screen, press G for Graphics Editor.

Drawing the Figure:

1. Press the I key 10 times (the cursor moves upward in small steps)
2. Press the D key (draws a line)
3. Press the greater-than key (>), remember to hold down the shift key as you do so.
4. Press the I key three times (cursor moves in larger steps)
5. Press the J key twice (cursor moves left)
6. Press the F key (draws a rectangular frame)
7. Press the O key (the letter, not the zero) five times
8. Press the S key (cursor skips to a new starting position)
9. Press the M key seven times
10. Press the J key four times (moves the cursor inside the figure)
11. Press the C key (draws on oval)

Coloring the figure:

12. Press the I key three times and the K key once (moves the cursor inside the oval)
13. Press the 1 key (one) and then the A key
14. Press the comma key six times and the S key
15. Press the J key nine times and the M key three times
16. Press 2, then B

Naming the figure:

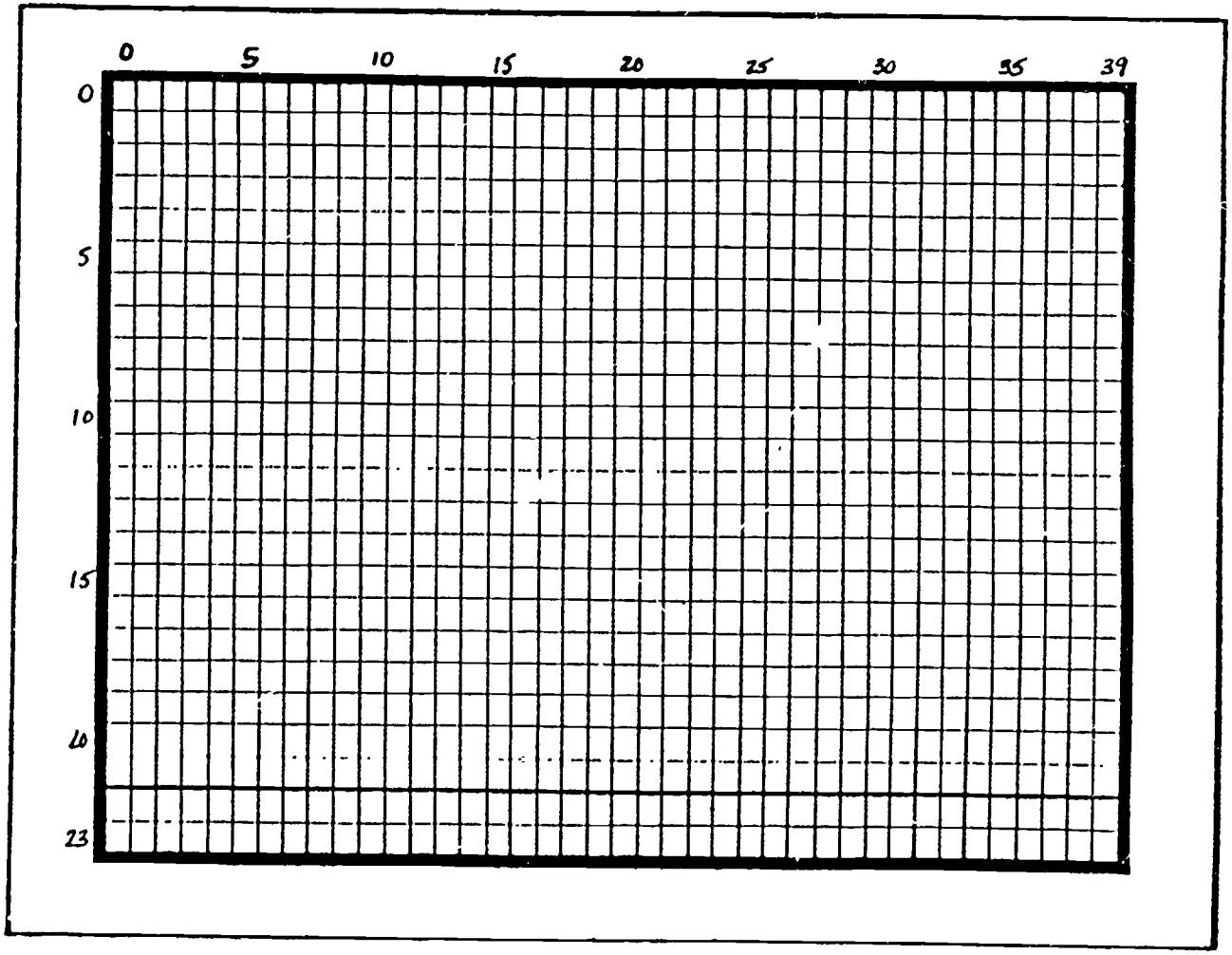
17. Press I and K keys once each
18. Press the T key (allows you to type on the figure)
19. Type Doodles then press the ESCape key

Quit the edits by pressing the Q key

Page No. ___ Title _____

Author _____

Date _____



Center for Innovation in Teaching the Handicapped
Indiana University

*** SOFTWARE LOAN AGREEMENT ***

In borrowing the microcomputer software specified below, I acknowledge that I have read the provisions for its care and use and agree to abide by those provisions. I also verify that I understand the stated consequences for the violation of these provisions.

-SuperPILOT Editor's Manual
-SuperPILOT Language Reference Manual
-SuperPILOT Author Disk
-SuperPILOT Lessons Disk
-SuperPILOT Log Disk

Copy Number:_____

Loan Provisions

1. This program is copyrighted and copy protected. I will not copy this program nor attempt to copy it.

2. I will not write in either of the manuals, nor will I remove the reference cards in the back of each manual.

3. I will exercise due care in the use of this program and will ensure that the program disk is not damaged in any way. I will operate the Author Disk at all times with a write-protect tab correctly in place.

Consequences

1. I understand that I am liable for prosecution under the federal copyright law if I copy the above software.

2. If I lose this software package, or the Author Disk only, I must pay $125.00, which is the cost of replacing the program.

3. If I lose either of the manuals, or if I write in either of the manuals, I will pay $25.00 for each one that must be replaced.

4. I will not be held liable for normal wear and tear on the manuals or the storage box for the program.

Any fees owed will be deducted from the $300.00 stipend. I have read the above conditions, understand them, and agree to abide by them. I have inspected the program materials and have found them to be in good condition with the following exceptions:

Signature	Date Borrowed	Date Due

Name:

Colors:

0 = Black	4 = Black2
1 = Green	5 = Orange
2 = Violet	6 = Blue
3 = White	

Part 1: (20 points) Give a brief description of what each of these commands do:

1. TS:ES

2. TS:F2

3. TH:

4. D:W$(10)

5. A:R

6. W:4

7. AS:

8. PR:L

9. J:@A

10. C:N=N+1

Part 2: (28 points) Complete these commands:

1. T:Name one thing that you can wear on your hands.
   A:
   M:

   (Correct answer: gloves or mittens)

2. T:What day comes after Tuesday?

   (Write the command that would allow one misspelling in the student's answer)

3. Write the commands that would provide feedback and then go on to the next question:

   *QUES1
   TS:ES
   T:What is the capital of Indiana?
   A:
   M:Indianapolis

```
*CORRECT
T:You got it right!
W:2
E:
*WRONG
T:You got it wrong.
W:2
E:
*QUES2
TS:ES
T: . . .
```

4. Write the command that would start this routine over if the student typed a letter instead of a number:

```
*PROBLEM1
T:What is 3 + 5?
A:#s

M:8
```

5. Write a subroutine that would erase lines 5 through 10 on the screen:

*

6. If you wanted to keep count of the student's correct answers in a variable named g, what command would have to precede the first question?

7. The computer gives the student 10 problems and counts his correct answers with a variable called r. Write the command that would send the student to a subroutine called END if the student's final score is greater than 8.

Part 3: (8 points)

1. Which command tells the screen to print orange characters?

   a. TS:B5

   b. TS:F5

   c. TS:B1

   d. TS:F1

2. Which commands would you use to make your whole screen green?

   a. TS:ES
      TS:B1

   b. TS:ES
      TS:F1

   c. TS:F1
      TS:ES

   d. TS:B1
      TS:ES

3. Which is the appropriate command, A or B?

   T:Did you enjoy the SuperPILOT class?
   A:
   M:n
   A) TY:So sorry to hear that.
   B) TN:So sorry to hear that
   M:y
   .
   .
   .

4. Circle ALL of the responses which would be judged correct using the following match command: m:%24%

   a. 24

   b. I am 24 years old.

   c. I am 24.

   d. 2244

Part 4: Short Answer (12 points)

1. What is the difference between the U: and the J: commands?

2. What does the command TN3: mean?

3. Write the command to create a variable named "w" which is a random number between 0 and 4.

4.
```
d:x$(10)
d:y$(10)
c:x$="mother"
c:y$="father"
```

Write a command which would print this line (using the appropriate variable):

He is my father.

AUTHORING SYSTEMS

VANILLA PILOT (Comm. 64, disk, $29.95, Fisher Scientific) A disk-based authoring system for the Commodore 64 microcomputer. Allows non-programmers to write CAI software for student use. Turtle graphics and color/sound may be added to lessons.

VANILLA PILOT (Comm. 64, tape, $29.95, Tamarack Software) A tape-based authoring system for most Commodore microcomputers. Just a few commands allow creation of interactive courseware without learning a programming language. Keyboard graphics may be included and you can switch between dark type on a white screen and the standard light type on a dark background.

COMMODORE PILOT (Comm. 64, disk, $59.95, Fisher Scientific) With this state-of-the-art version, you can design lessons using the color, sound, and graphics capabilities of the Commodore 64. These include characters you design (for math, science, or foreign language.), sprites (colorful moveable objects for animation), hi-resolution graphics, and music synthesis to add interest to textual lessons. A "Run Only" version is included so students can use the program, but not rewrite it.

TRS-80 Author 1 (TRS-80: I & III, disk, $149.95, Radio Shack) This screen oriented authoring system features full screen editing, graphics, branching, score keeping, hints, and more.

CAT (Computer-Aided-Teaching) (TRS-80: I, III, & 4; disk, $39.95, DynaComp, Inc.) An authoring package consisting of two programs, CATGEN and CAT. CATGEN is used by the teacher to create lessons, tests, and lesson plan files. CAT is used by the student to run the lessons and tests created by the teacher. A lesson created by CATGEN consists of a screen of mixed text and graphics with an optional practice question mode. The teacher may also enter up to 50 multiple choice questions; automatic correction and scoring is provided by CAT during administration of the test. The lesson plan file creation procedure provides the teacher with a method of automatically administering a specific sequence of lessons and tests to the student. The lesson plan consists of a chain of lesson files and test file names, which will automatically be run by CAT when the lesson plan is entered.

EUREKA LEARNING SYSTEM (Apple II, disk, $495. Eiconics) The Text Writer portion of the system allows the author to create text materials of their choice. An Illustrator File contains special math, music and other symbols for inclusion in the lesson. A Shape Editor allows the inclusion of graphics. Fee quoted is for license on one computer system. License for additional systems is $49.50 each. Extensive support and future revisions are included.

APPILOT II (Apple II, II+, disk, $99.95, Follett) Educators can easily create multiple copies of lessons for student use. This program features high- and low-resolution graphics, a built-in student timer, Symtec light pen output, mixed high-resolution graphics and text, and voice output using the Muse Voice program. The program is complete with a comprehensive user's guide and interactive lessons to teach Appilot programming. Please note: The above listed item is not recommended for use on an Apple IIe microcomputer.

THE AUTHOR (Apple II, 2 disks, $325, Follett) This authoring system requires no computer or programming skill. The teacher generates lessons through menu-driven screens. Lessons can provide immediate or delayed reinforcement, branch learners automatically to additional instruction, keep detailed records of learner performances, and permit learner easy access to any part of a lesson for review/practice. A detailed user's manual and complete self-instructional course are included.

VOICE BASED LEARNING SYSTEM (Apple II with Shadow/VET, disk, $99.95, Sterling Swift Publishing Co.) The Voice Based Learning System allows a student to use voice or sound to input answers to specially created instructional material. The program is an authoring system that allows the creation of individualized material without learning a programming language. Voice entry is through the Shadow/VET, produced by Scott instruments ($995.00). The voice entry system is trained to recognize each individual's speech or repeatable sounds.

MICROTEACH (Apple II, Atari 800, 2 disks, $275, Compumax) This two disk system is comprised of the Teachers Aide, which allows creation of the courseware, and Student Pak, which allows the student to run the courseware without being able to modify it. Text may be presented as pages, or the text may be scrolled. The time a page is displayed is controlled by the author. The system always provides menus of available options. It is available in English or Spanish.

E-Z PILOT (Apple II, disk, $49.95, Teck Associates) The twelve commands in E-Z PILOT allow a person to create complete interactive courseware on any subject. Color, sound and large size alphabets can also be used. Graphics or other programs in BASIC can be inserted anywhere in the course materials.

Z.E.S. CAI SYSTEM (Apple II, 3 disks, $249, K-12 Micro Media) This versatile lesson-authoring system features full lesson creation/amendment capabilities; color hi-res graphics and animation routines; individual and class reporting and progress monitoring; completely menu-driven programs, and such user-friendly aids as automatic error checking, field delimiters, and prompting. The Z.E.S. CAI System consists of a 114-pp user's manual, question input sheets and three disks that allow teachers to create and amend lessons, incorporate graphics, and store lessons and student records--all without any programming knowledge required.

SOFCRATES: The Courseware Creator (Apple II, disk, $199, SVE) Lessons created with the Sofcrates authoring system feature colorful, multi-font text and high-resolution graphics on the same screen. questions that allow for more than one correct answer and provide automatic remediation, student progress tracking, animations with any one of over 50 characters, an index for quick retrieval of any lesson page or information, and color background options. Lesson formats can vary to include tutorials, drill and practice, question/information, and testing.

THE AUTHOR (Apple II+, 2 disks, $195, SVE) Self-instructional, interactive computer lessons can be created with this authoring package. Lesson types may be drill and practice, tutorial, or simulation; questions may be multiple choice, fill-in-the-blank, sequencing, or true/false. Record-keeping function keeps track of learner's performance. Users manual and convenient lesson planning sheets are included.

PUBLISHERS OF LESSON AUTHORING SYSTEMS

Compumax, Inc.
P.O. Box 7239
Menlo Park, CA 94025
(415) 854-6700

Eiconics, Inc.
P.O. Box 1207
211 Cruz Alta Road
Taos, NM 87571
(505) 758-1696

Dynacomp, Inc.
1427 Monroe Avenue
Rochester, NY 14618

Fisher Scientific
EduMart Computer Division
1458 N. Lamon Avenue
Chicago, IL 60651

Follett Quality Courseware
4506 Northwest Highway
Crystal Lake, IL 60014
(800) 435-6170

K-12 Micro Media
172 Broadway, Dept. CL
Woodcliff Lake, NJ 07675

Radio Shack (Education Div.)
1600 Tandy Center
Fort Worth, TX 76102
(817) 390-3302

SVE-Society for Visual Education, Inc.
Dept. VB, 1345 Diversey Pkwy.
Chicago, IL 60614

Sterling Swift Publishing Co.
1600 Fortview Road
Austin, TX 78704
(512) 444-7570

Tamarack Software
P.O. Box 247
Darby, MT 59829
(406) 821-4596

Teck Associates
P.O. Box 8732
White Bear Lake, MN 55110
(612) 429-5570

TIES
1945 W. County Road B2
St. Paul, MN 55113
(612) 633-9100

MICROCOMPUTERS IN SPECIAL EDUCATION

MODULE 6: BASIC Programming

In the beginning there were no real programming languages, no real computer memory devices as we know them today. There were only wires to connect in a certain way to cause the computer to execute certain instructions. If the wires were connected in the correct order, the computer could solve a problem. Even though the connection of all the wires was tedious and error-prone, it was still very exciting to see the computer perform a relatively complex calculation at such an increased speed over human effort. But there had to be a better way, and there was.

Through the combined efforts of John Mauchly, J. Presper Eckert (developers of the ENIAC computer) and John Von Neumann, the first computer memory device was developed. Suffice it to say that it was based on some of the same principles that govern the operation of the picture tube in your television set. For the first time it was possible to store the data and the instructions for a program in the computer. Other developments at about the same time allowed the computer's processor to extract and execute each one of a series of instruction codes from its memory in order. These instructions, of course, had been placed in the computer's memory by the programmer. As a result, it became feasible to create and store a series of machine language instructions (a program) which, when executed by the processor, could translate another program (stored in the computer memory as English-like text) into the very simple machine language instructions that it understands. Later on, the processor could be told to execute the program it had just translated. This process of translation can also be called compilation or assembly: hence the terms compiler program or assembly language program. The English-like text referred to above was actually the beginning of what we know today as programming languages.

There are a multitude of programming languages available for writing programs for today's computer systems, both big and small. Some examples of these are: BASIC, PASCAL, FORTRAN, ALGOL, ADA, MODULA-2, FORTH, LISP, SIMULA, SNOBOL-4, and COBOL. These languages have been developed to accomplish different kinds of tasks with the computer or to better express the solution of a task with the computer. BASIC and PASCAL (named for the mathematician Blaise Pascal), for example, were developed in part to teach the concepts of programming to students. Some languages are intended to be general purpose. This means that the languages have commands and facilities which should be adequate to produce reasonably efficient programs to solve a wide variety of problems. FORTRAN, BASIC, PASCAL, and ADA are general purpose languages. Other languages are special purpose. This means that the language has been developed to produce programs which are geared to a more limited group of tasks, such as business data processing (which deals heavily with computer files of information). COBOL (COmmon Business-Oriented Language) is an example of a language designed to handle business data processing tasks. SIMULA is a language designed with features that allow programs to be written which simulate real-life sequences of events in order to study the interactions of those events on one another.

Even though they are capable of having many different programming languages developed and made available for them, today's microcomputers usually come with BASIC (Beginner's All-purpose Symbolic Instruction Code) as a 'built-in' language. In many cases, the BASIC language interpreter program has been placed in a ROM (Read-Only Memory) integrated circuit chip by the manufacturer of the computer. BASIC was developed under the direction of John Kemeny and Thomas Kurtz at Dartmouth College in the mid-1960s. It was designed to be easy to learn and implement so that a wide variety of disciplines could utilize it for learning and problem-solving purposes.

BASIC is usually implemented as a language interpreter rather than as a language compiler. It was noted above that a compiler program translates your program (written in the language in question such as BASIC or PASCAL) into the appropriate simple machine-language instructions **once**, pointing out any errors you have made as it translates. If the program does not contain any errors (or at least the compiler program cannot find any), your program may then be executed. An interpreter program, on the other hand, extracts a small segment of your program (usually a single command) and (if it does not find any errors) translates the single command into a small series of machine-language instructions which will accomplish only that small segment of your program. That small segment is then immediately executed and the interpreter program regains control and extracts the next small segment of your program. This process is similar to a human interpreter who listens to a sentence that you say and makes the proper translation for the person you are speaking with (who does not speak your language).

There is great debate in the computer field about which language is 'best' to program in or teach programming with. Many program authors have a favorite language they like to use. Recently (circa 1984-1985), the authors of BASIC have modified the definition of the BASIC programming language to include some of the most useful features of other languages, most notably PASCAL. The authors call this new language 'True BASIC.' Indeed, this new implementation of an old standard does squelch some of the critics of the old BASIC. Nevertheless, BASIC (new or old) remains a viable language with which to explore simple programming concepts.

Objectives

At the end of this module, participants will be able to:

1. Describe the major commands and functions in BASIC.

2. Demonstrate the ability to enter, edit, and execute a previously written simple BASIC program.

3. Produce and debug a simple BASIC program using at least 10 of the commands covered in instruction.

4. Identify the general areas of differences in programming in BASIC on different computers.

5. Describe the advantages and disadvantages of the fact that the BASIC programming language is non-standard.

TRS-80 Model I

1. Make certain there is no disk in disk drive unit #0.
2. Make certain that the pushbutton on the back-right of the keyboard is pushed in and that the power indicator light on the keyboard is lit.
3. Place a TRSDOS System Disk (which has the BASIC programming language on it) into disk drive unit #0.
4. Press in and release the RESET button on the back left edge of the keyboard. Disk drive unit #0 should start and the message 'DOS READY' should appear on the screen.
5. Type: BASIC<ENTER>
6. To the prompt of: 'How Many Files?', press the <ENTER> key.
7. To the prompt of: 'Memory Size?' (or 'Mem Size?'), press the <ENTER> key.
8. BASIC will prompt you that it is ready with:
   READY
9. You are now in the BASIC programming environment.
10. You may leave BASIC by typing: CMD "S"<ENTER> and you will return to 'DOS READY'. Make certain you have SAVED any program you might be working on to a diskette before you leave!

TRS-80 Model III/4

1. If the computer is OFF, turn it on and wait a moment or two.
2. Place a TRSDOS System Disk (which contains the BASIC programming language) into disk drive unit #0.
3. Press the orange RESET button (located in the upper right of the keyboard. In a moment you will be prompted to enter the current date. Type the date in the format shown on the screen and press the <ENTER> key. The next prompt allows you to set the time of the internal clock of the computer. You may skip setting the time by pressing the <ENTER> key in response to the prompt.
4. The message 'TRSDOS Ready' should be present on the screen.
5. Type: BASIC<ENTER>
6. To the prompt of: 'How Many Files?', press the <ENTER> key.
7. To the prompt of: 'Memory Size?' (or 'Mem Size?'), press the <ENTER> key.
8. BASIC will prompt you that it is ready with:
   READY
   >
9. You are now in the BASIC programming environment.
10. You may leave BASIC by typing: CMD "S"<ENTER> and you will return to 'TRSDOS Ready'. Make certain you have SAVED any program you might be working on to a diskette before you leave!

TRS-80 Level II BASIC Development Commands
Abbreviated List

While most microcomputers allow the user to develop new programs in BASIC, different brands of microcomputers (and, sometimes, different models within the same brand) use somewhat different commands to access and modify those programs. Therefore, simply knowing how to program in BASIC is not sufficient; the user must also be familiar with the machine-specific commands used to save, load, and edit programs.

The commands which follow are used on the TRS-80 Models I, III, and 4 to control development and modification of BASIC programs:

NEW<ENTER>

Erases current program from RAM and tells BASIC that a new program is about to begin. Note: before entering NEW you should SAVE any program in RAM that you may wish to access later.

Example:
>NEW<ENTER>

EDIT <line #><ENTER>

Causes the instruction on the specified line number to be displayed and allows programmer to edit that line.

Example:
>EDIT 100<ENTER>

In the example, BASIC is instructed to display the instruction contained in line 100 and to wait for further instructions on editing the line. The following list contains editing commands which facilitate changes:

C — Indicates user's desire to change the next character to the right of the cursor. Typing a number before C allows user to change that number of characters to the right. For example, 3C allows the user to change the next 3 characters after the cursor.

L — When entered after a change, lists the rest of that line and restarts the cursor at the beginning of the line.

S — Allows user to search for the first occurrence of the next character typed. This is a quick method for moving the cursor rapidly through a long line.

D — Deletes the next character to the right of the cursor. Typing a number before D deletes that number of characters to the right. For example, 5D deletes the next 5 characters after the cursor. BASIC confirms each deletion by displaying the deleted characters bracketed by exclamation points (e.g., !PRINT!). Entering L after a deletion causes BASIC to display the new line as it now appears.

I Allows user to insert new characters to the right of the cursor. Pressing the SHIFT and keys simultaneously ends the insert and returns to the normal edit mode.

Q Lets the programmer quit the edit mode without saving changes just made on the current line. Used when the programmer changes his/her mind.

E Ends the edit and saves changes just made on the current line. BASIC returns to the Command Mode.

<ENTER> Ends the edit and saves changes just made on the current line. Lists new line with changes reflected.

A line may also be edited without typing EDIT <line #>. Simply type the line number and the new instruction for that line, and press <ENTER>. BASIC will insert the line into the program, writing over the old instruction for that line.

Example:

former line:	100 PRINT "HELLO"
at Command Mode type:	100 PRINT "GOODBYE"<ENTER>

In the example, BASIC replaces the former instruction at line 100, PRINT "HELLO", with the new instruction, PRINT "GOODBYE".

==========

DELETE <beginning line number>-<ending line number><ENTER>

Permits you to delete the statements contained on <beginning line number> to <ending line number>. Note that it is also possible to effectively replace a statement on a line number by simply retyping the line number you wish to replace and retyping the statement or statements on that line.

Example:

>DELETE 10-100<ENTER>

In the example, the statements on lines 10 through 100 would be deleted from the program. If you wished to replace the entire line on line number 10 only, then you could simply type:

>10 <new statement><ENTER>

and the statements on line number 10 would be replaced by those that you type in anew.

==========

LIST <beginning line number>-<ending line number> <ENTER>

Causes the statements on and between <beginning line number> and <ending line number> to be displayed on the screen for inspection. If both the beginning and ending line numbers are omitted, then the entire program is displayed. If this mode is selected and it is desired to pause the display, press the SHIFT key down and hold it down while simultaneously pressing the '@' (at-sign) key. To resume the listing, repeat this

procedure. To abort the listing function, press the BREAK key and the machine will return to Command Mode.

Example:
LIST<ENTER>
LIST 10-100<ENTER>

In the first example above all of the lines of the program would be listed on the screen. In the second example, the statements on line numbers 10 through 100 would be listed on the screen.

==========

LLIST <beginning line number>-<ending line number><ENTER>

This is a variation on 'LIST' above, but instead of the program being displayed on the screen, the statements will be listed on the printer provided it is connected and ready to print.

Example:
>LLIST<ENTER>
>LLIST 10-100<ENTER>

In the first example above, ALL of the lines of the program would be printed on the printer. In the second example, just lines 10 to 100 would be printed.

==========

SAVE "<file name>"<ENTER>

Same as the 'LOAD' command below but stores the program you have been working on, on the specified disk unit with the name of <file name>.

Example:
>SAVE "MYPROG:0"<ENTER>

In the example, the program which is currently in BASIC will be stored with the name 'MYPROG' on disk drive unit number 0.

==========

LOAD "<file name>"<ENTER>

Causes the Disk Operating System program to 'bring in' the program from the disk to BASIC so that it may be changed, added to, or run.

Example:
>LOAD "MYPROG:0"<ENTER>

In the example, BASIC has been instructed to load the program called 'MYPROG' from disk drive unit number 0. Note that the program must have been previously SAVEd on the disk unit or a 'File Not Found' error will occur. If the error does occur, however, you will be returned to the '>' prompt.

==========

RUN<ENTER>

Causes a program already in memory (such as a program you've just edited) to run. To stop a running program, press the BREAK key, and the machine will return to the command mode.

Example:
>RUN<ENTER>

RUN "<file name>"<ENTER>

Causes a program not presently in memory to be loaded from the disk into memory and run. Press the BREAK key to stop a running program and return to Command Mode.

Example:
>RUN "MYPROG:1"<ENTER>

In the example, BASIC has been instructed to load the program called 'MYPROG' from disk drive unit number 1 and to run that program.

==========

CMD "S"<ENTER>

Allows programmer to leave BASIC and return directly to TRSDOS. This command works on TRS-80 Models I, III, and 4. The programmer should save any BASIC program currently in memory (if the program is a new one or has been edited) before using this command.

==========

CMD "D:<disk drive #>"<ENTER>

This a high level command specific to the TRS-80 Models III and 4 only. It is used to display the directory (catalog of files) on the specified disk drive WITHOUT leaving BASIC.

Example:
>CMD "D:0"<ENTER>

In the example, BASIC is instructed to display the list of files contained on disk drive 0. Following the display, BASIC returns to the Command Mode.

TRS-80 Level II BASIC Programming Commands
Abbreviated List

==========

REM <comment>

This command, which is an abbreviation of the word 'REMARK', is used to insert documentation and comments into the program. Any text which follows this command is ignored by BASIC.

Example:

```
10 REM ****************************************
20 REM * Program Name: STATES                  *
30 REM * Program Author: IMA TEACHER           *
40 REM * This program queries the student's    *
50 REM * knowledge of state names, capitals,   *
60 REM * and general geographic location       *
70 REM * within the United States              *
80 REM ****************************************
```

In the example above, it can be discerned from the remarks what the name of the program is, who authored it and what function it performs. Adequate documentation using the REM command is always a good programming practice.

==========

DIM <variable name>

Declares the <variable name> to be an array with certain dimensions.

Example:

```
10 DIM A$(5)
20 DIM A(5,10)
```

In the example, the variable A$ is an array which will hold up to 5 strings of characters; the variable A is an array with 5 rows and 10 columns of numbers. Each item in the arrays is accessed by giving the correct number of the string, as in: A$(3) or A(2,3)

==========

CLS

The command CLS clears the screen of the TRS-80.

Example:

```
10 CLS
20 END
```

==========

LET <variable name> = <variable name> or <constant> or <expression>

Assigns the value of the right-hand side of the '=' to the variable on the left-hand side. The ARITHMETIC OPERATORS are:

+ for Addition
- for Subtraction
* for Multiplication
/ for Division

Example:
```
10 LET A = 3*6
20 LET B = 5/A
30 LET A$ = "HELLO"
```

In the example, the word LET in each assignment statement is optional, so the equivalent statements would be:

```
10 A = 3*6
20 B = 5/A
30 A$ = "HELLO"
```

==========

GOTO <line number>

Causes the next statement to be executed to be the statement found on <line number>.

Example:
```
10 PRINT "HELLO"
20 GOTO 10
```

In the example, the word 'HELLO' would be printed on the screen until the 'BREAK' key is pressed on the keyboard.

==========

GOSUB <line number>
RETURN

Causes the next statement to be executed to be the statement found on and after <line number> until a RETURN command is encountered (see below).

Example:
```
10 PRINT "HELLO"
20 GOSUB 100
30 PRINT "I HAVE RETURNED FROM THE SUBROUTINE"
40 END
100 PRINT "I AM NOW IN THE SUBROUTINE"
110 RETURN
```

In the example, a subroutine is called in line 20 which at some point finishes its task and executes a RETURN command. After the RETURN command is executed the next statement to be executed is line 30, which is immediately after the GOSUB.

==========

FOR <counter variable> = <starting value> TO <stopping value>

Starts a program loop, each iteration of which increments the value of the <counter variable> from the <starting value> by 1 until it reaches the value of the <stopping value>. All statements between the 'FOR' statement and the 'NEXT' statement are executed once for each iteration of the loop.

Example:
```
10 FOR I = 1 TO 10
20 PRINT "HELLO"
```

```
30 NEXT I
```

In the example, the word 'HELLO' will be printed on the screen 10 times. When the 'NEXT' statement is encountered, the value of the <counter variable> is incremented by 1 and if the value is less than or equal to 10, then the statement at line number 20 is executed again.

==========

IF <condition> THEN <statement> ELSE <statement>

The <condition> will be tested for validity and if it is true, the statement following the 'THEN' will be executed; otherwise the statement following the 'ELSE' will be executed. A <condition> has two arguments, which may be variables or constants. The two arguments are separated by a CONDITIONAL OPERATOR. The conditional operators are:

>	(is greater than)
<	(is less than)
=	is equal to
<>	is not equal to

Example:

```
10 IF A < 1 THEN LET A = 1 ELSE GOTO 10
```

In the example, if the variable A has a value which is less than 1, then the variable A will be assigned the value of 1; otherwise the program will resume with statement number 10. Please note that the 'ELSE' clause of the IF command is optional.

==========

INPUT "<prompt message>"; <variable name>

Causes the <prompt message> to be displayed on the screen followed by a '?' (question mark). If the variable is a character string variable, then whatever is typed on the keyboard before the <ENTER> key is pressed will be assigned as the value of the string variable. If the variable is a number variable then only numbers will be allowed to be input.

Example:

```
10 INPUT "What is your age"; AG
20 INPUT "What is your name?"; N$
```

==========

PRINT <variable name> or <constant>

Causes the value of the variable or constant to be displayed on the screen.

Example:

```
10 PRINT "The answer is: "; N
20 PRINT "Student name:"; N$
```

In the example, the semicolon in each PRINT statement prevents the normal addition of a carriage return at the end of the printing of the first item to be printed. This causes the message "The answer is:" to appear on the same line as the answer itself.

```
10 REM ****************************************
20 REM * Program Name: GRADEBK                *
30 REM * Author: Yura Teacher (Ima's sister,  *
40 REM * Purpose: This program summarizes      *
50 REM * student assignment scores and assigns a *
60 REM * letter grade according to a weighting  *
70 REM * value for each assignment towards the  *
80 REM * final grade. The assignment description *
90 rem * data and the student score data are    *
100 rem * contained in BASIC DATA statements at *
110 rem * the end of the program.               *
111 rem ****************************************
120 CLEAR 1024:DEFINT A-Z
121 CLS
122 PRINT "GRADEBK - Student Grade Calculation Program"
123 PRINT
1000 DATA 5
1001 DATA 15,15,10,20,20
1002 DATA .1,.1,.1,.3,.4
1003 DATA 5
1004 DATA "JOHN JONES",      08,10,10,18,16
1005 DATA "CATHY MASTERS",   12,13,06,18,18
1006 DATA "JAN SELLERS",     14,12,08,15,19
1007 DATA "DAVID KELLY",     15,12,05,18,17
1008 DATA "MARY PETERS",     11,13,12,14,17
```

CIMS
COMPUTER-BASED IEP MANAGEMENT SYSTEM
VERSION 6.0

Software Development: Robert Eckert

INTRODUCTION

The Computer-Based IEP Management System (CIMS) is a microcomputer-based software system for creating, managing, and reporting individual educational programs (IEPs) through the use of instructional objectives. All CIMS software is interactive and CIMS has the capability to be linked to a school system's central computer facility where the collected data can be aggregated to monitor and report group data.

HARDWARE REQUIREMENTS

CIMS is developed for the TRS-80 Model I and III/4 (48K, dual disk drive).

SOFTWARE REQUIREMENTS

CIMS consists of three interrelated systems, each stored on separate disks. The Objectives Subsystem manages the library of instructional objectives and transfers to a student's IEP the objectives that you select. The IEP Subsystem stores student profiles and associated IEPs, and enables you to update this information and to obtain several types of progress reports. The Utilities Subsystem provides facilities for managing files of information used in CIMS and for transmitting CIMS information between different locations.

OPERATION

Running CIMS

(a) Turn on the machine.
(b) Insert the disk labeled with the system you wish to use (Objectives, IEP, or Utilities) in drive 0.
(c) Press the reset button.
(d) When you receive the TRSDOS Ready prompt, type Do CIMS <enter>.
(e) The password to enter all programs is IEP.

The Objectives Subsystem

The Objectives Subsystem enables you to build and modify a library of objectives, which are then available for transfer to student IEPs. To create this subsystem, you first define courses/areas (e.g. Mathematics, Reading, Consumer Business). You then create a list of long range objectives (LROs) and short range objectives (SROs), with accompanying criterion levels, for each course/area.

Upon accessing the Objectives Subsystem, you will be presented with the following command menu:

<1> TRANSFER Objectives to Student IEP
<2> ACCESS Objective Courses/Areas
<3> ACCESS Objectives
<4> HELP
<5> EXIT

The first command, transfer objectives to student IEP, enables you to select and transfer a single objective or a group of objectives from the Objectives Subsystem library to a student IEP. The second command, access objective courses/areas, enables you to add to, delete from, display, or print the contents of the Objectives Subsystem's Course(s)/Area(s) File. The third command, access objectives, permits you to add, edit, delete, display, or print the contents of the Objectives Subsystem library file. The fourth command, help, provides you with information about how to use the Objectives Subsystem. The final command, exit, returns you to TRSDOS.

IEP Subsystem

The IEP subsystem permits you to manipulate and report information contained in a specific student IEP. Upon accessing the IEP subsystem, you will be presented with the following menu:

<1> ACCESS Student IEP
<2> REPORT Student IEP
<3> REPORT Student IEP Progress
<4> ACCESS Student Profile
<5> HELP
<6> EXIT

The first command, access student IEP, enables you to add, edit, or delete a student's IEP. You may also update a student's IEP through this command, by providing CIMS with information regarding the student's attainment of specific IEP objectives. The second command, report student IEP, permits you to generate a printed listing of objectives contained in an IEP. The third command, report student IEP progress, enables you to generate an IEP progress report for a student, listing all completed objectives and summarizing percentage of objectives completed for each LRO and each Course/Area. A sample of a CIMS IEP progress report is appended to this document.

The fourth command, access student profile, permits you to create and/or update demographic information regarding an individual student. NOTE: YOU MUST CREATE A STUDENT PROFILE BEFORE YOU CAN CREATE AN IEP FOR THAT STUDENT. The fifth command, help, provides you with information about using the IEP Subsystem. The final command, exit, returns you to TRSDOS.

The Utility Subsystem

The Utility Subsystem contains various commands to permit you to transfer information from one CIMS system to another and from one CIMS

location to another. Upon accessing the Utility Subsystem, you will be presented with the following menu:

<1> TRANSMIT/RECEIVE Student IEP(s) to/from Remote CIMS Site
<2> PURGE Student IEP Files from Student IEP Disk
<3> ASSIGN School Identification t CIMS Subsystem Disk
<4> COPY Course/Area File (Obj. Subsys. to IEP Subsys.)
<5> HELP
<6> EXIT

The first command, transmit/receive student IEP(s) to/from remote CIMS site, enables you to copy student profiles and IEPs from a student IEP disk and to transmit this information to another CIMS site. Conversely, this function also enables you to receive student profiles and IEP(s) from another CIMS site. The second command, purge student IEP files from student IEP disk, enables you to purge (erase) student profiles and IEP files. The third command, assign school identification to CIMS subsystem disk, enables you to assign a CIMS IEP Subsystem or Utilities Subsystem disk to a particular school. The fourth command, copy course/area file, enables you to copy the contents of the course/area file on a CIMS Objectives Subsystem disk to a CIMS IEP Sybsystem disk. This procedure is necessary when transferring objectives to a student IEP.

The fifth command, help, provides you with information about how to use the Utilities Subsystem. And finally, exit, returns you to TRSDOS.

CIMS 6.0 Student IEP Progress Report
Individualized Educational Program (NOT a Contract) Page 1
ALAN ADDER IEP CONFERENCE 3/84

Student Number/IEP Number/Name: 1223335789-01/1/ ALAN ADDER
Address: 124 COMPUTATION LANE, ALPHA, NEW YORK 44331
School: BLUE RIVER HIGH SCHOOL
Grade: 10
Program/Service: LD/PT4
Date of Last Psychological Test: 01/15/73

Current Mathematics Instructional Level: .....

Current Reading Instructional Level: .....

The courses/areas and related objectives in this student's individualized education program have been developed specifically for pupils formally in the above listed special education area (Mildly Mentally Handicapped, Learning Disabled, Visually Handicapped, etc.).

At the end of the semester, the teacher will indicate which of the objectives have been mastered in a report which will be included in the pupil' records. This achievement report will reflect not only the student's individu achievement in the course/area, but will also be utilized to plan his/her futu course/area objective assignments.

It also should be noted that the teacher of each course/area may assign additional objectives, should student progress warrant it.

The achievement of the objectives shall be determined

by:.....

Person(s) attending Case Conference:

.....

.....

.....

.....

.....

.....

.....

.....

.....

CIMS 6.0 Student IEP Progress Report
Individualized Educational Program (NOT a Contract)
ALAN ADDER IEP CONFERENCE 3/84

Student Number/IEP Number/Name: 1223335789-01/1/ ALAN ADDER

B21 :Consumer Business II :
LRO: 1.0 The student will understand consumer concepts.

SRO: 1.0 Defines a "contract".

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 2.0 Reads a contract and briefly explains its contents.

* COMPLETED * Criterion: 70 percent - Score: 100 percent
SRO: 3.0 Defines the term "warranty".

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 4.0 Explains the benefits of a warranty.

* COMPLETED * Criterion: 70 percent - Score: 100 percent
SRO: 5.0 Tells how to lodge a product complaint.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 6.0 Names local and national consumer protection agencies.

* COMPLETED * Criterion: 75 percent - Score: 80 percent
SRO: 7.0 Explains the functions of small claims court.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 8.0 Explains the importance of reading instruction booklets.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 9.0 Explains how to locate a repair service.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 10.0 Selects medical, legal, and financial professionals.

* COMPLETED * Criterion: 70 percent - Score: 80 percent
SRO: 12.0 Explains the use of credit cards.

* COMPLETED * Criterion: 80 percent - Score: 100 percent
SRO: 14.0 Defines the term "layaway".

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 16.0 Explains how to find an apartment and what is involved in renting.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 17.0 Explains what is involved in buying a car.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
>>>> 78 % of the SROs for LRO 1.0 have been COMPLETED.
>>>> 78 % of the SROs for B21 have been COMPLETED.

BEHV:Behavioral Objectives :

CIMS 6.0 Student IEP Progress Report
Individualized Educational Program (NOT a Contract)
ALAN ADDER IEP CONFERENCE 3/84

Student Number/IEP Number/Name: 1223335789-01/1/ ALAN ADDER

LRO: 1.0 Attends class regularly.

SRO: 6.0 Organizes assignments effectively.

* COMPLETED * Criterion: 90 percent - Score: 90 percent
SRO: 7.0 Completes assigned tasks.

* COMPLETED * Criterion: 90 percent - Score: 95 percent
>>>> 67 % of the SROs for LRO 1.0 have been COMPLETED.
>>>> 67 % of the SROs for BEHV have been COMPLETED.

MATH:Mathematics Objectives (7-12)
LRO: 8.0 The student will improve in the understanding of money.
SRO: 1.0 Instantly recognizes all coins.

* COMPLETED * Criterion: 100 percent - Score: 100 percent
SRO: 2.0 Recognizes one, five, ten, and twenty dollar bills.
* COMPLETED * Criterion: 100 percent - Score: 100 percent
SRO: 3.0 Makes correct change after a purchase up to $25.00.
* COMPLETED * Criterion: 80 percent - Score: 80 percent
SRO: 8.0 Reads and solves story problems using money.

* COMPLETED * Criterion: 80 percent - Score: 95 percent
SRO: 9.0 Tells the approximate value of such items as food, clothing, rent, and cars.
* COMPLETED * Criterion: 80 percent - Score: 90 percent
SRO: 12.0 Tells why it is important to get a receipt for purchases.
* COMPLETED * Criterion: 80 percent - Score: 100 percent
SRO: 13.0 Explains the value of standard hourly work rates.

* COMPLETED * Criterion: 80 percentSRO: 15.0 Explains the purpose of tax.
* COMPLETED * Criterion: 80 percent - Score: 100 percent
SRO: 16.0 States the purpose of Social Security.

* COMPLETED * Criterion: 80 percentSRO: 17.0 Knows the cost of postage letters and post cards.
* COMPLETED * Criterion: 80 percent - Score: 100 percent
SRO: 18.0 Fills out an application for a savings account.

* COMPLETED * Criterion: 80 percent - Score: 100 percent
SRO: 19.0 Fills out deposit and withdrawal slips.

* COMPLETED * Criterion: 80 percent - Score: 100 percent
>>>> 57 % of the SROs for LRO 8.0 have been COMPLETED.
>>>> 57 % of the SROs for MATH have been COMPLETED.

V10 :P.V.E. Related

CIMS 6.0 Student IEP Progress Report
Individualized Educational Program (NOT a Contract)
ALAN ADDER IEP CONFERENCE 3/84

Student Number/IEP Number/Name: 12233 3789-01/1/ ALAN ADDER

LRO: 2.0 Can use self-help organizations to maintain a job.

SRO: 1.0 States five reasons for a person losing a job.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 2.0 States ten things to do to maintain a job.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 3.0 States five factors which could aid in promotion.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
SRO: 5.0 Can use self-help organizations to maintain a job.

* COMPLETED * Criterion: 75 percent - Score: 100 percent
>>>> 80 % of the SROs for LRO 2.0 have been COMPLETED.
>>>> 80 % of the SROs for V10 have been COMPLETED.

MODULE 7: RESOURCE MATERIALS

Technology and Special Education: A Resource Guide

M. Nell Bailey & Sharon L. Raimondi

The rapid proliferation of microcomputers in education has been paralleled by the emergence of related journals, associations, user groups, and other resources. To assist special educators in reviewing this growing list of materials and resources, the Office of Special Education Programs, U.S. Department of Education, provided funding for Project EduTech. Among the ongoing activities of this project is the maintenance of an information base focusing on the applications of technology to special education, as well as the dissemination of that information to educators and other interested groups.

This article presents a selection of resources, periodicals, books, organizations, projects, and networks designed specifically for special educators. In some cases, resources of a general nature have been included because of their usefulness and applicability to special education.

Periodicals

Subscription rates for periodicals are for the Continental United States only, and are current as of January 1984. Subscription rates for other countries may be slightly higher.

Catalyst. Western Center for Microcomputers in Special Education, 1259 El Camino Real, Suite 275, Menlo Park CA 94025. $12.00/6 issues, individual subscription. The Catalyst is intended to interpret, clarify, and communicate the latest microcomputer research, developments, products, and applications to special education users.

Closing the Gap. P.O. Box 68, Henderson MN 56044. $15.00/6 issues. This newsletter explores the uses of computers (including peripherals and software) with the handicapped and special education students. Special modifications and applications for the deaf and hearing impaired, blind and visually impaired, mentally handicapped, learning disabled, and severely physically handicapped are also addresed.

Communication Outlook. Artificial Language Laboratory, Computer Science Department, Michigan State University, East Lansing MI 48824. $12.00/4 issues. This quarterly newsletter on electronic aids for the handicapped is a publication of the International Society for Augmentation and Alternative Communication, and is published jointly by the Artificial Language Laboratory and the Trace Research and Development Center at the University of Wisconsin.

The Computing Teacher. 1787 Agate Street, University of Oregon, Eugene OR 97403. $21.50/9 issues. Published by the International Council for Computers in Education (ICCE), this journal is geared toward persons interested in the instructional use of computers. It emphasizes teaching about computers, computer applications, teacher education, and the impact of computers on curriculum.

Journal of Special Education Technology. Exceptional Child Center, Utah State University, UMC 68, Logan UT 84322. $17.00/4 issues. Directed primarily to administrators, researchers, and teachers, this journal publishes information, research, technology reviews, and reports on innovative applications of educational technology furthering the development and education of exceptional children.

School Microcomputing Bulletin. Learning Publications, Inc., P.O. Box 1326, Holmes Beach FL 33509. $28.00/10 issues. Microcomputing trends and educational computing applications are reported in this Bulletin, which also includes information on sources of materials, software evaluations, workshops, and special field reports by educators.

Teaching, Learning, Computing. Seldin Publishing, Inc., 1061 South Melrose, Suite D, Placentia CA 92670-7180. $24.00/10 issues. TLC is a new magazine geared toward classroom teachers who are interested in personal computing. Readers are kept up-to-date on developments in computer legislation, special education, administrative planning, and the academic disciplines. Each issue is planned to include current computer trends and predictions; indepth product reviews; computer management techniques; software test results and evaluations; and profiles of innovative educators.

Books

Computers for the handicapped in special education and rehabilitation: A resource guide. Eugene: University of Oregon, Rehabilitation and Training Center, 1982. This resource guide includes more than 180 annotated citations from 39 journals, over half dated since 1980. The entries focus on computer-assisted instruction in both an educational and client setting and computer-managed instruction. Available from ICCE, 1787 Agate Street, University of Oregon, Eugene OR 97403.

Dominguez, J., & Waldstein, A (Eds.), Educational applications of electronics technology. Monmouth OR: WESTAR, 1982. This collection of articles assembled by WESTAR staff is designed to enhance the reader's awareness of developments, issues, and applications of electronic technology in education.

Goldenberg, E.P. Special technology for special children: Computers to serve communication and autonomy in the education of handicapped children. Baltimore MD: University Park Press, 1979. The use of LOGO with cerebral palsied, deaf, and autistic students is described, as well as an innovative view of computer use in special education.

Hagen, D. Microcomputer resource book for special education. Reston VA: Reston Publishing Company, 1983. Available from The Council for Exceptional Children (CEC). This book provides an understanding of the microcomputer as a life competency tool, and shows how computers can work for children at home and in the classroom. The full spectrum of software and adaptive devices is described. Disabilities are looked at, one at a time, and the computer needs of each disability group are examined. The advantages and disadvantages of each type of program are weighed. A series of appendices provide information about more than 200

publishers of software products. Products are grouped by disability area and detailed information is provided about each program's use.

Personal computers and the disabled: A resource guide. Cupertino CA: Apple Computer, Inc., 1983. This resource guide was prepared as a public service document to stimulate research into personal computer applications for the disabled. The guide features articles on how the computer is helping the disabled to overcome obstacles and to deal with their limitations. A section on products for special needs, a list of organizations, and a section citing over 60 resources are included. A copy is available free by writing to Apple Computer, Inc., 20525 Mariani Avenue, Cupertino CA 95014.

Taber, F.M. Microcomputers in special education: Selection and decision-making process. Reston VA: The Council for Exceptional Children, 1983. The author provides special educators with an introduction to the microcomputer and its uses in both general and special education. Six chapters cover subjects ranging from software considerations and evaluation to media selection and elementary programming for the microcomputer. Each chapter includes a summary, a list of sources for more information, and bibliography.

Organizations

The sharing and exchange of information and ideas with fellow members can reduce time spent in the search for technology news and information. Organizations specifically concerned with technology and the handicapped, as well as others that have special interest groups in this area, are listed in the following section. Local user groups are extremely popular and provide an excellent resource for educators. Seek out computer user groups in your community.

Association for the Development of Computer-Based Instructional Systems (ADCIS). ADCIS Headquarters, Miller Hall 409, Western Washington University, Bellingham WA 98225. Individual non-affiliate membership fee: $40.00/year.

ADCIS is an international not-for-profit association for professionals in the field of instructional technology. This association facilitates communication between product developers and users to reduce repetitive efforts among developers of CAI materials. ADCIS provides a variety of membership services including annual conferences, workshops, CBI publications, and local chapter affiliations. It also sponsors several special interest groups, including Educators of the Handicapped.

Association for Special Education Technology (ASET). P.O. Box 152, Allen TX 75002. Membership fee: $25.00/year.

ASET, a national affiliate of the Association of Educational Communication and Technology, was established for the following purposes: to bring together disciplines sharing a common interest in improving the use of technology in special education; to identify and publicize unique instructional needs of special education students; and to promote improved federal legislation for technology in special education. Membership in ASET includes the quarterly publication, The Journal of Special Education Technology, four issues of ASET Report, presentations at national conventions, and an ongoing forum for members

to focus on those needs of handicapped students which can be assisted through improved technology.

Computer-Using Educators (CUE). P.O. Box 18547, San Jose CA 95158. Individual membership fee: $8.00/year.

This membership organization is committed to expanding the use of computer technology in education. Initiated in California, CUE has expanded to include members in 44 other states and 12 foreign countries. Membership includes a subscription to the CUE Newsletter.

The Illinois Council, Congress of Organizations of the Physically Handicapped, Committee on Personal Computers and the Handicapped (COPH-2). 2030 Irving Park Road, Chicago IL 60618. Membership fee: $8.00/year.

The purpose of this organization is to search out, evaluate, and share information on hardware, software, software modifications, educational materials developed for disabled people, and use of computers as part of the personal development of handicapped children. Members benefit from hardware and software demonstrations, computer loans, technical assistance, a membership list (ENTER-ACT), a quarterly publication (Link and Go), the testing, manufacture, and distribution of low-cost, computer-related hardware, and all-day meetings every other month.

Michigan Association for Computer Users in Learning (MACUL). P.O. Box 628, Westland, MI 48185. Membership fee: $5.00/year.

MACUL is an organization of Michigan educators that coordinates instructional computing activities throughout the state and sponsors a yearly conference. Founded in 1975, its membership is now 6,000 (10% out-of-state) and includes a subscription to the MACUL newsletter. MACUL also has a small collection of programs for the Apple II, the PET, and the Atari 800. The cost of each diskette is $10.

Technology and Media (TAM). Dr. Charles McArthur, Membership Chairperson, University of Maryland, Institute of Exceptional Children and Youth, College Park, MD 20742. Membership fee: $10.00/year.

TAM, an international association of special education professionals interested in technology and media, was recently organized by members of The Council for Exceptional Children (CEC). Formal application has been made by this organization to become a Division of CEC. TAM is an organization for professionals, parents, handicapped persons, and members of the business community who are concerned with the impact of technology and media upon the diagnosis, treatment, and educational rehabilitation of exceptional persons. Membership dues support the development and dissemination of a journal and newsletter which contain information on the diverse topics related to technology and media in special education.

The Young People's LOGO Association (YPLA). 1208 Hillsdale Drive, Richardson TX 75081. Membership fee: $9.00/year (under 18); $25.00/year (adults).

YPLA is an independent, nonprofit national computer club run for and by young people. A subscription to Turtle News, a monthly magazine, is included with membership. Members also have access to a software exchange, an electronic bulletin board, and a resource library. The software exchange, for all popular personal computers, includes teacher

and user developed software ranging from simple to complex games and educational and business software. Also available is software developed by and for the handicapped.

Resource Groups

A growing number of specialized groups address specific aspects of computer use. Those listed here will be of special interest to educators.

Artificial Language Laboratory. Michigan State University, Computer Science Department, East Lansing MI 48824.

The Artificial Language Laboratory is involved in basic research in the field of computer processing and formal linguistic structure. Research includes speech analysis and synthesis, interspecific communication, pattern recognition of human electromyographic (EMG) signals, and neurolinguistics. The Laboratory is also involved in developing vocational and educational aids for the blind.

Center for Computer Assistance to the Disabled. P.O. Box 314, Hurst TX 76053. No membership fee.

This independent, nonprofit organization is less than a year old. It provides a location where disabled persons can gain assistance with the application of microcomputers. This organization encourages the use of off-the-shelf hardware, currently available technology, and products as a means of aiding the education and training of mentally and physically handicapped persons.

MicroSIFT. Northwest Regional Education Laboratory, 300 S.W. Sixth Avenue, Portland, OR 97204.

MicroSIFT is a clearinghouse for descriptive and evaluative information about microcomputer instructional software and teacher information. Provided in print form, this information is distributed to state and local education agencies and selected commercial and professional periodicals.

Minnesota Educational Computing Consortium (MECC). 3490 Lexington Avenue, North, Saint Paul, MN 55112.

MECC is a nonprofit publisher of training materials and educational courseware for most popular personal computers. MECC can help any teacher or parent interested in putting personal computers in the classroom. The Consortium has also directed a project to develop computer learning packages for use in science, mathematics, and social studies courses. Network, a bimonthly instructional newsletter listing available materials, as well as MECC's catalog, are free upon request.

National Center for Research in Vocational Education. 1960 Kenny Road, Columbus, OH 43210.

Materials on various aspects of career and vocational education, including programs for special needs populations, are available from the Center. Also available are several publications including a discussion guide on microcomputers in vocational education, an administrator's guide to microcomputer resources, and a system for evaluating microcomputer courseware for vocational and technical educators.

SOFTSWAP/CUE. Microcomputer Center, SMERC Library, San Mateo County Office of Education, 333 Main Street, Redwood City, CA 94063.

SOFTSWAP is a joint project of the San Mateo County Office of Education and Computer-Using Educators (CUE); it serves as a clearinghouse of public domain educational software. Programs are available free of charge to educators who copy them onto their own disks at the Microcomputer Center, or may be ordered for $10.00. SOFTSWAP is also a software exchange. Any educator who contributes an original program on a disk may request any SOFTSWAP disk in exchange. BLOCKS, the courseware development system developed at the California School for the Deaf, and other courseware are available from this source.

Technical Education Research Centers, Inc. (TERC). Computer Resource Center, 8 Eliot Street, Cambridge, MA 02138.

TERC is a nonprofit, educational research corporation that has been in existence 16 years. TERC's mission is to study social and technological changes and to apply that knowledge to improving education. Two centers currently constitute TERC--the Special Needs Center and the Technology Center. TERC also publishes a newsletter entitled Hands On.

Trace Research and Development Center. 314 Waisman Center, University of Wisconsin-Madison, 1500 Highland Avenue, Madison, WI 53705.

In cooperation with the Communications Aids and Systems Clinic of the University of Wisconsin Hospitals, this Center studies and develops techniques and aids to augment communication skills of clinical patients. The Center collects, documents, and disseminates information on these and other communication aids and techniques.

Networks

Electronic networks for special educators are a new and rapidly growing resource which provides subscribers an opportunity to receive and share professional information about special education. This service involves the use of computers which are linked to s central information center.

SpecialNet. National Association of State Directors of Special Education, 1201 16th Street, N.W., Suite 404E, Washington, DC 20036. (202/822-7933).

SpecialNet provides current information and electronic mail capability for persons interested in services and programs for the handicapped. Over 1500 users in all 50 states subscribe to SpecialNet, and 50 national and state bulletin boards provide information relating to special education that ranges from federal news to technology. Several state departments of special education have established state bulletin boards and now use SpecialNet exclusively to support in-state networking systems. SpecialNet is available to anyone who has access to a computer terminal or microcomputer.

The Handicapped Educational Exchange (HEX). 11523 Charlton Drive, Silver Spring, MD 20902. Computer access phone: 301/593-7033. For information about the exchange, call 301/681-7372.

HEX is a free national computer network devoted to the exchange of ideas and information on the use of advanced technologies to aid

handicapped indviduals. This information bank includes source listings of computer-assisted instructional materials, software, and machinery for the handicapped. It can be assessed via telephone (and modem) or TDD.

The Electronic Information Exchange System (EIES). New Jersey Institute of Technology, Newark, NJ 07102. (201/645-5503).

EIES is a national computer conferencing system and information network that hosts many conferences, one of which is EIES/Handicapped. This conference is designed for exchanging information concerning disabled persons.

Projects

The growing support of technology at federal and state levels is reflected in the increase in the number of technology projects being funded. Three projects funded by the Office of Special Education Programs (SEP) are described in this section, as well as three that are funded at the state level.

Please be aware that there are many states conducting projects on technology and the handicapped. To list them all would be a monumental task. For information about other states who have unique and exemplary projects in special education, consult the Training and Model Exchange Project (EC 160811, 58pp.), a publication prepared annually by the Council of Administrators of Special Education, Inc. (CASE), a Division of CEC. (Order from ERIC Document Reproduction Service, P.O. Box 190, Arlington, VA 22210.)

Project C.A.I.S.H. (Computer Assisted Instruction and Support for the Handicapped). 3450 Gozio Road, Sarasota FL 33580.

Project C.A.I.S.H. is a Title VIB project funded through the State of Florida to provide hardware and software for all exceptionalities. The project provides teacher training, consultation for implementation workshops, using the Apple II Plus and Apple IIe microcomputers exclusively. The first year of the project was devoted to the physically handicapped; the second year to the mentally and emotionally handicapped. Currently, the project is addressing the remaining exceptionalities with appropriate hardware and software. An interim report is available free from the FDLRS Clearinghouse, Bureau of Education for Exceptional Students, Division of Public Schools, Florida Department of Education, Knott Building, Tallahassee FL 32301.

Project EduTech. JWK International Corporation, 7617 Little River Turnpike, Annandale VA 22003.

Project EduTech is designed to provide technical assistance to state and local education agencies in the appropriate use of technology in special education. Ongoing activities include selecting widespread special education issues on which to focus each year, developing reports and other modes of disseminating information on technological advances that may help resolve these issues, and maintaining an informative system on technological advances.

Market Linkage Project for Special Education. LINC Resources, Inc., 3857 North High Street, Suite 225, Columbus OH 43214.

Through its contact with the Office of Special Education (OSE) of the U.S. Department of Education, the Market Linkage Project for Special Education offers product developers assistance in understanding how to develop educational products to meet national distribution standards and to assure that these products are available in the classroom. Initiated in 1977, it has assisted dozens of projects to get several hundred products into the marketplace. Any project funded by ED/OSE can take advantage of this program, which gives free technical assistance, helps develop your products, and assists in locating a publisher.

Microcomputers in the Schools--Applications in Special Education. SRA Corporation, 901 South Highland Street, Arlington VA 22004.

In a joint effort with the Cosmos Corporation, this two-year project will analyze the experiences of local schools currently using microcomputers with programs for the handicapped, and then develop and disseminate useful, research-based information to assist administrators and educators in introducing microcomputers in special education programs.

Project RECIPE (Research Exchange for Computerized Individualized Programs of Education). 1001 South School Avenue, Sarasota FL 33577.

Project RECIPE is a nationally validated instructional management system designed to help educators create and implement individualized education programs (IEPs). The RECIPE system includes instructional objectives, a criterion-referenced assessment system, and instructional strategies. A management component is designed to track and record student progress. The system provides progress reports to parents and annual progress reports. It may be used at the classroom, building, or district level by teachers or administrators to store and manipulate student demographic and program data, as well as to fulfill the requirements of IEP and progress reporting.

SECTOR Project. Exceptional Child Center, UMC-68, Utah State University, Logan UT 84322.

The SECTOR project is a state-funded special education computer technology resource located at Utah State University. SECTOR conducts overviews of adaptive and communication devices, reviews of courseware for handicapped students, software for special education management, and maintains a bibliographic information base for special education. The Center is also conducting research on the interactive videodisc.

REFERENCES

Bailey, M. N. & Raimondi, S. L. (1984). Technology and special education: A resource guide. Teaching Exceptional Children, 16(2), 273-277.

Cohen, V. B. (1983). Computer courseware development and evaluation: Criteria for the evaluation of microcomputer courseware. Educational Technology, 23(1).

Jay, T. B. (1983). The cognitive approach to computer courseware design and evaluation. Educational Technology, 23(1), 22-26.

MODULE 8: CONCLUSION

K500 MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION
CRITERION TEST 1

Instructions: Some multiple-choice items on this test may be unlike any others you have encountered. These are designed to test the breadth and accuracy of your knowledge rather than your ability to recognize generally correct information. A "*" next to these test items indicates that one, any combination, all, or none of the answers provided on this test item may be correct. On these questions you should circle the letter(s) beside any or all of those answers that you think are correct. Unstarred multiple-choice items on this test follow the traditional format in which only one answer is correct; on these, simply circle the letter adjacent to the answer you select.

1. Define and briefly explain the following terms:

ROM:

RAM:

CPU:

Disk Drive:

Monitor:

2. List at least three major cautions to follow in caring for floppy diskettes.

*3. The major generations of computers are associated with which of the following technological advances:

A. vacuum tubes
B. large-scale integrated circuitry
C. geranium diodes
D. integrated circuitry
E. silicon resistors
F. transistors

4. List three ways computer-based information can be permanently stored.

5. "Higher level" languages were developed for use with computers primarily because:

   A. They made computer operations faster.
   B. They were easier for computers to translate.
   C. They were easier for programmers to use.
   D. They were needed to guide more complex computer operations.
   E. They were needed to operate "peripherals."

6. True or False:

   T F A "16 bit machine" refers to the fact that one byte for that computer equals 16 1,0 code units.

   T F The major advantage of a "16 bit machine" is in the speed by which information can be read and transmitted by the machine.

   T F "64 K RAM" means that there is 64,000 kilobytes of memory available to use for generating and storing programs.

   T F Information stored in ROM is erased when the machine is turned off.

   T F The term "digital computers" refers to the way in which information is coded and handled in the machine.

   T F Most software will not run on machines made by various computer manufacturers, but usually software that runs on one model of machine made by a particular company will run on other models made by the same company.

7. List three examples of computer peripherals.

8. The diskette containing the actual software programs for running on a particular computer is called a:

9. A system that allows several users to work on a computer at once is called:

10. An educational application in which the computer is used to assess students' level of incoming information, to give progress tests, and to offer information concerning instructional resources is referred to as what type of program:

A. Computer Managed Instruction (CMI)
B. Tutorial Program
C. Computer-based Instruction (CBI)
D. Computer Assisted Instruction (CAI)
E. Simulation

MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION

1984 Institute Evaluation

We are very interested in your feelings regarding the usefulness of the workshop/course in which you have just participated. Your feedback will be used in evaluating the institute and in revising its content for future presentations.

* *

I. Please rate each module presented in the institute on a scale from A (excellent) to F (poor). In addition, please indicate the modules' greatest strengths and greatest weaknesses from your point of view.

MODULE 1: INTRODUCTION TO MICROCOMPUTERS (history of computers, general structure and operations of microcomputer, hands-on experience with ComputerWRITER/DOODLE)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 2: SOFTWARE APPLICATIONS (computer terminology, types of educational software applications, hands-on experience with CITH software--CMMRS, Spellmaster, CRIS, CIRIS)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 3: INTRODUCTION TO WORD PROCESSING (applications of word processing in special education, hands-on experience with Word Handler)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 4: EVALUATING EDUCATIONAL SOFTWARE (general research findings on CBI, evaluation criteria, hands-on experience evaluating educational software)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 5: INTRODUCTION TO AUTHORING SYSTEMS (authoring system applications, hands-on experience with SuperPILOT)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 6: INTRODUCTION TO BASIC PROGRAMMING (discussion of major BASIC commands, hands-on experience with BASIC programming)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 7: LOCAL COMPUTER RESOURCES (Indiana Clearinghouse for Computer Education, visit to local computer store)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 8: GENERAL ISSUES (hardware and software resources, packaged vs. teacher-developed software, issues and logistical considerations)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

* *

II. Looking back on the institute/course as a whole, please rate the following aspects of instruction on a scale of A (excellent) to F (poor).

	RATING
1. Usefulness of printed handouts	_____
2. Clarity of presentations by instructors	_____
3. Comprehensiveness of presentations	_____
4. Order of module presentation (sequencing)	_____
5. Pacing of instruction	_____
6. Instructors' level of preparedness	_____
7. Helpfulness of instructors	_____
8. Value of "hands-on" experiences	_____
9. Value of field experiences	_____
10. Usefulness of this experience in relation to your present teaching situation	_____

* *

III. Please feel free to use the space below to add any additional comments you may have about any aspect of the institute/course.

* *

COURSE EVALUATIONS

1984 Microcomputer Applications in Special Education Course

OVERVIEW

Recent advances in computer technology have major implications for the career and job functions of the special education teacher. Microcomputers have been demonstrated to be effective tools enabling special educators to: provide handicapped students the opportunity for experience in computer use and programming; accurately assess and assist in remediating students' basic skills; develop Individual Education Plans (IEP's); monitor student academic performance; and provide feedback and reports to students, teachers, administrators, and parents. However, because of a lack of training, the majority of Indiana's special education teachers are not able to take advantage of the full potential microcomputers offer.

In order to provide special education teachers with sufficient resources and experience in the use of this technology with their students, the Department of Special Education and The Center for Innovation in Teaching the Handicapped (CITH) at Indiana University conducted a teacher training course in microcomputer applications. The course was offered on the Bloomington campus during the summer of 1984. A total of 20 participants from the State of Indiana were enrolled and received over 56 hours of direct instruction in the form of lectures/demonstrations, discussions, and "hands-on" experiences. Additional "hands-on" experience was provided through 30 hours of unstructured open-lab sessions.

Course evaluation results indicate that participants

1) showed significant gains in knowledge related to the structure, operation, and classroom applications of microcomputers,

2) demonstrated positive gains in attitudes toward educational applications of microcomputers and in personal confidence in ability to implement this new technology in the classroom, and

3) rated the course very highly regarding usefulness of the experience in relation to their present teaching situations.

GOALS AND OBJECTIVES

A considerable amount of attention is being given to the role of the microcomputer for educating students. With school districts throughout the nation investing large amounts of money into purchasing hardware and software for classroom use, it will soon become mandatory for teachers to be skilled in the use of microcomputers. As in many areas of education, the microcomputer appears to have a bright future in the field of special education. Some possibilities or applications of microcomputer technology in special education include administrative data and record keeping, instruction, classroom-based data collection and analysis, and development of IEP's. This course was designed to prepare special education teachers to use and evaluate microcomputers and microcomputer software and to gain a familiarity with a variety of computer instructional programs.

Specifically, it was expected that at the end of this course each participant would be able to:

a. Identify major milestones in the history of computers.

b. Define various computer-related terms.

c. Differentiate between various types of computers and their uses.

d. Identify various types of hardware and peripherals.

e. Connect and operate various types of computer hardware.

f. Provide a rationale for educational applications of computers.

g. Operate/use various educational software programs.

h. Evaluate various educational software packages using appropriate criteria.

i. Demonstrate how to perform major functions in operating a word processing program.

j. Use a word processing program to write a project report.

k. Demonstrate how to perform major functions in operating a lesson authoring system.

l. Use a lesson authoring system to produce a short software lesson for special education students.

m. Identify various types of commands in Basic.

n. Copy and operate a simple program in Basic.

It was anticipated that attainment of these objectives would enable teachers to use microcomputers accurately and efficiently assess students' present levels of performance and develop IEP's to deliver appropriate instruction on an individual basis and to monitor and report student academic performance. Further, the course was intended to equip teachers with skills necessary for providing students with training in computer literacy and basic microcomputer programming using an authoring system.

COURSE DESCRIPTION

The teacher training course was offered on Indiana University's Bloomington campus during the summer of 1984. Participants attended three 1/2-hour sessions on two afternoons a week for eight weeks. Presentations were in the form of lectures/discussions, demonstrations, and "hands-on" laboratory experiences. Additional open-lab, "hands-on" experience was available for three hours prior to class on each of 10 days during the course. Instructors were members of the Special Education Department's graduate faculty assisted by experienced professional staff members of the Center for Innovation in Teaching the Handicapped.

Sufficient numbers of two major brands of micro-computers, Apple II and Radio Shack TRS-80, were availed for use by the participants. In addition, information was availed regarding other major brands to which the teachers might have access and differences among models discussed when appropriate.

Selection of Course Participants

Prior to the course, the availability of the course was announced through a series of brochure mailings during the last two weeks of April. Each public school building principal and each Director of Special Education in the State of Indiana received the announcement and was asked to make application forms available to their teachers. The course was open to all special education teachers currently employed in Indiana's public schools, particularly those with little or no previous experience in classroom applications of the microcomputer. To maximize individual attention and access to the computers, enrollment was limited to 20 participants.

Forty four applications were submitted and twenty teachers were selected through use of a stratified random sampling procedure which controlled for school districts in which applicants were employed. This procedure insured that one school district would not be over-represented by the selected participants. A list of 11 alternates was also generated in the same manner. Briefly, the teachers represented 16 different school districts across the state. Twelve participants served elementary students, four served middle/junior high students, and four served high school students. The majority of participants reported having had little or no computer training (0-10 clock hours) previous to the course. Only four reported more than 10 hours of previous training (ranging from 12 to 30 hours.)

Instructional Content

The intent of the instructional staff was to provide participants with exposure to and experience in a wide range of microcomputer applications in special education. The curriculum content was, therefore,

divided into eight modules, each covering a separate but related topic. Topics presented and instructional activities included:

MODULE 1: Introduction to Microcomputers (2 sessions)

- Administration of pre-assessment materials (attitude scale and mastery test)
- Discussion of the history of computers and the place of microcomputers in that history
- Introduction to the general structure and operation of microcomputers
- Hands-on experience with simple word processing/computer graphics software designed as a student motivational aid in the classroom

MODULE 2: Software Types, Applications, and Sources (2 sessions)

- Presentation of microcomputer-related terms
- Discussion of types of software, their applications and sources
- Hands-on experience with educational software dealing with remedial math and reading
- Hands-on experience with software designed to assist special education teachers with instructional planning

MODULE 3: Introduction to Word Processing (2 sessions)

- Presentation of word-processing concepts
- Discussion of word-processing applications in the special education classroom

- Hands-on experience using "Word Handler," a word-processing system for APPLE's

MODULE 4: Evaluating Educational Software (2 sessions)

- Discussion of criteria to use in evaluating commercial software designed for instructional use in special education settings
- Hands-on experience using and evaluating commercial software designed for instructional use

MODULE 5: Using Authoring Systems (4 sessions)

- Presentation on uses of lesson authoring software
- Presentation and software tutorial on using the SuperPILOT lesson authoring system
- Hands-on experience creating SuperPILOT lessons oriented to a CAI application in special education

MODULE 6: Introduction to BASIC (2 sessions)

- Presentation on functions of major commands in the BASIC programming language
- Hands-on experience copying and running BASIC programs
- Hands-on experience creating simple BASIC programs

MODULE 7: Hardware and Software Resources (1 session)

- 0 Presentation by the Indiana clearinghouse for computer education staff on services available to Indiana teachers
- 0 Visit to a local commercial computer facility to gain familiarity with additional microcomputer brands and products

MODULE 8: Current Issues in Educational Microcomputing (1 session)

- 0 Discussion of available hardware and software resources suitable for special education
- 0 Discussion of advantages and disadvantages of packaged teacher-developed software
- 0 Discussion of issues involved and logistical considerations in the integrating of computers in classroom environments
- 0 Administration of post-assessment materials (attitude scale and mastery test)
- 0 Closing remarks and workshop evaluation by participants

1984 COURSE EVALUATION

Evaluation of the 1984 microcomputer course consisted of three components: 1) assessment of changes in teacher attitudes toward microcomputers as a result of participating in the course; 2) determination of teacher mastery of the content presented; and 3) assessment of consumer satisfaction with the training experience. Results of these evaluation components are discussed below.

Changes in Teachers' Attitudes

Immediately prior to and following participation in the course, teachers' attitudes toward computers in education were assessed through the use of an attitude scale developed and field-tested by the staff. The attitude scale contains a series' of demographic questions followed by 30 statements to which teachers indicate their degree of agreement or disagreement by circling the appropriate number on a 6-point scale (1 indicates strong agreement; 6 indicates strong disagreement.) The demographic questions and list of 30 statements included on the attitude scale may be found in Appendix A.

Comparison of teachers' mean responses showed change (more than 0.3 points on the 6-point scale) in attitudes toward 14 of the statements as a result of participation in the course. Those statements which reflected a mean change of more than 0.3 points are displayed in Table 1. The lower the mean rating of statements in Table 1, the more positive the teachers felt toward the concept reflected by the statement.

- - - - - - - - - - - - - - -

Insert Table 1 about here

- - - - - - - - - - - - - - -

Table 1

Attitude scale statements reflecting a change in teachers' attitudes toward microcomputers as a result of participation in the course.*

A. Statements for Which a POSITIVE Change in Attitude Occurred

Item #	Statement	Mean Ratings (N=20) Pre	Post	Change
21	I can write microcomputers programs to meet the special needs of my students.	4.00	2.50	-1.50
3	Errors made by computers could have disastrous effects on our society.	3.60	4.47	+0.87**
30	I am confident I can choose appropriate microcomputer programs for my students.	2.50	1.65	-0.85
27	I am not afraid that I will damage the microcomputer or it programming.	2.36	1.75	-0.61
6	I am confident of my ability to learn to use microcomputers effectively in the calssroom.	1.85	1.26	-0.59
26	Humans have sufficient control over computers to prevent ill effects of their use in our society.	2.82	2.40	-0.42
18	I am not worried that someone will steal or vandalize a microcomputer that I am in charge of.	3.50	3.20	-0.30

B. Statements for Which a NEGATIVE Change in Attitude Occured

Item #	Statement	Mean Ratings (N=20) Pre	Post	Change
10	All students should be taught to program computers.	3.55	5.05	+1.50
23	The available instructional program adequately branch students to higher and lower instructional levels when necessary.	3.73	4.65	+0.92
17	Most available instructional programs are age-appropriate for use with my students.	3.64	4.45	+0.81
11	Most of the software programs avaliable are easy for students to use without help from the teacher.	3.25	4.05	+0.80

Item #	Statement	Mean Ratings (N=20) Pre	Post	Change
13	I am confident I will receive (or did receive) adaquate training before and during my initial use of the microcomputer in my classroom. (prior to institute).	3.33	3.85	+0.52
2	I am pleased with the variety of programs available for use with special education students.	4.45	4.95	+0.50
4	Computer-based instruction motivates students to learn material in which they wouldn't ordinarily be interested.	1.80	2.16	+0.36

* Each statement was rated on a 6-point scale (1 = strong agreement; 6 = strong disagreement.) The lower the mean rating the more positive the teachers as a group felt toward the statement.

**An exception to the above statement--since the item was negatively stated, a higher mean rating indicates a more positive attitude toward computers' effects on society.

Teachers' responses indicated a positive change in attitude toward seven of the attitude scale statements. The majority of these statements (items 21,30, 27, and 6) dealt with personal confidence issues. As a result of the course, participants apparently were more confident of their ability to write software programs, choose commercially available programs, and use microcomputers effectively in special education classrooms. Responses to two statements (3 and 26) reveal that teachers also perceive effects of computers on society in general to be less negative than they judged prior to the course.

Responses to seven other statements indicate a negative change in teachers' attitudes as a result of the course. Four of these items (statements 23, 17, 11, and 2) are related to the quality of available instructional software and its suitability for use with special education students. Clearly, after exposure to evaluation criteria and a representative sample of currently available software, the teachers became more discerning and less satisfied with the instructional value of available software. The inadequecies of software reviewed may also contribute to teachers' slightly negative change in attitude toward the motivational quality of computer-based instruction (statement 4). The greatest negative change came with statement 10, "all students should be taught to program computers." A mild disagreement with this statement prior to the course changed to more than a moderate disagreement after the teachers were exposed to BASIC programming techniques. Apparently they were most convinced that special education students would not be able to master this skill.

The 16 statements to which teachers showed little change in attitude dealt primarily with the educational benefits of computer-assisted

instruction to students and teachers. It should be noted, however, that the teachers already had moderately positive attitudes regarding these issues.

Mastery of Course Content

Pre-and posttest versions of an information mastery test were administered prior to and following the course, respectively. Copies of the two versions of the mastery tests may be found in Appendix B & C.

The two versions of the mastery test were devised to measure participants' knowledge of microcomputer-related terms, microcomputer structure and operations, and software-related information. Both forms consisted of true-false, multiple-choice, matching, and short-answer questions. The two versions covered the same content, but the posttest version was designed to be somewhat more rigorous in terms of the transfer of knowledge and discrimination of answers required of the participants.

The average score attained by participants on the mastery pretest was 46.6%. The average score on the posttest was 75.6%, resulting in an average gain of 29 percentage points as a result of participation in the course. This gain is statistically significant ($p<.001$) despite the increased difficulty of the mastery posttest. Primary gains were in the areas of microcomputer-related terms and software-related information. Deficiencies were still notable in the area of microcomputer operations (item 6, posttest).

Consumer Satisfaction With the Course

Finally, immediately following the course, participants were asked to provide feedback regarding the usefulness of the experience. This feedback was collected through the use of the evaluation form which may be found in Appendix D.

The evaluation form asked participants to : 1) rate each of the eight modules on a scale from A (excellent) to F (poor) and to indicate each module's greatest strengths and weaknesses; 2) rate ten aspects of the course as a whole on the scale from A to F; and 3) provide additional comments which might aid in evaluation and/or revision of the course.

Participants' ratings of the eight modules and the course as a whole are displayed in Table 2.

- - - - - - - - - - - - - - - - -

Insert Table 2 about here

- - - - - - - - - - - - - - - - -

The modules on software applications and word-processing received the highest ratings of 3.70 (A = 4.00, F = 0.00), followed by the module on general issues related to computer-assisted instruction at 3.56, and the module on authoring systems at 3.50. The module on BASIC programming received the lowest rating, 2.79, primarily because participants indicated more time was needed to absorb instruction and practice the programming techniques.

As also shown in Table 2, all ten aspects of the course evaluated received a mean rating of above 3.00 on the four-point scale. Preparedness and helpfulness of the instructors were rated most highly at 3.89, with usefulness of the experience in present teaching position and value of "hands-on" experience following closely at 3.84 and 3.79, respectively. Pacing of instruction received the lowest rating of 3.05 with some participants indicating more time needed to cover some of the modules.

Not all teachers provided additional comments regarding the course (part III of the evaluation form.) All comments from those who did are listed below for the reader's benefit:

Table 2

Participants' ratings of the eight modules and the course as a whole

MODULE RATINGS MODULE:	Mean Rating *
1. Introduction to Microcomputers	3.15
2. Software Applications	3.70
3. Introduction to Word Processing	3.70
4. Evaluating Educational Software	3.25
5. Introduction to Authoring Systems	3.50
6. Introduction to BASIC Programming	2.79
7. Local Computer Resources	2.95
8. General Issues	3.56

GENERAL RATINGS	Mean Rating *
1. Usefulness of printed handouts	3.74
2. Clarity of presentations by instructors	3.58
3. Comprehensiveness of presentations	3.58
4. Order of module presentation (sequencing)	3.58
5. Pacing of instruction	3.05
6. Instructors' level of preparedness	3.89
7. Helpfulness of instructors	3.89
8. Value of "hands-on" experiences	3.79
9. Value of field experiences	3.16
10. Usefulness of this experience in relation to your present teaching situation	3.84

* 4.00 = A(excellent); 0.00 = F(poor).

Teacher 1: (1) It would have been helpful to have the printed handouts before need for them in class. (2) I was a "baby" in computer education. Some lectures were hard for me to follow. (3) I felt too much time was devoted to history and not enough time for the BASIC module. (4) The instructors were Super! (5) I'm a "learn by doing" person so the hands-on was most valuable.

Teacher 2: Enjoyable experience-good instructors.

Teacher 3: Thank you, I enjoyed this very much.

Teacher 4: (1) Sharing the computers for major projects were difficult. (2) Lack of help on days other than class days. (3) The instructors did a super job! You have someone special in Bob.

Teacher 5: This course has given me many valuable ides for school next year. It was a great introductory course.

Teacher 6: I greatly appreciate having had this opportunity. I hope more teachers do things like this. I don't think the short workshops (i.e., 3 days) are all than beneficial, and more in-depth study like this is excellent.

Teacher 7: I only hope this can be offered to more teachers in the future. It was a very well done and useful course.

Teacher 8: It would have been helpful to have received a printed list of participants, their specialties, where they work and/or live. The two authoring systems (PILOT and BASIC) seemed rushed for what was being asked of the student.

Teacher 9: Excellent class! Very, very helpful.

Teacher 10: This has been an extremely valuable experience for me. It has increased my knowledge and especially my self-confidence! If the same type of course is offered in the future - it should be 3 days a week instead of 2!

Teacher 11: The 3 of you came across as wanting to have us learn so that we can actually use computers, etc.

Teacher 12: It would have been beneficial to have a list of course members and information available: name, address, position, school system, etc., for future reference and consultation. I would like to have had in depth experience with the CITH-developed software.

Teacher 13: I am looking forward to sharing what I have learned here with my friends and fellow teachers. I feel that computers can be an effective tool for any teacher and especially for the Special Education Teacher. However, I feel that the software needs some kind of a monitoring system built into the programs so the teacher can keep a record of the students' progress and thus provide the best possible individualized program for each child.

APPENDICES

Appendix A: Attitude Scale

****

ATTITUDES: COMPUTERS IN EDUCATION

Bob Eckert, Sharon Goh, and Steve Bergman
Center for Innovation in Teaching the Handicapped
Indiana University, Bloomington, IN

****

GENERAL Information

A. What are the LAST 6 DIGITS of your Social Security Number?
(YOU WILL REMAIN ANONYMOUS)

_ _ _ _ _ _

B. Are you currently a student? (Please check ONLY one)
[ ] 1. No, I am not currently a student
[ ] 2. Yes, Parttime Undergraduate
[ ] 3. Yes, Fulltime Undergraduate
[ ] 4. Yes, Parttime Graduate Student
[ ] 5. Yes, Fulltime Graduate Student
[ ] 6. Other: _____

C. Are you currently a teacher? (Please check ONLY one)
[ ] 1. No, I am not currently a teacher
[ ] 2. Yes, Parttime Teacher in Sp. Ed. (Self-contained)
[ ] 3. Yes, Parttime Teacher in Sp. Ed. (Resource)
[ ] 4. Yes, Fulltime Teacher in Sp. Ed. (Self-contained)
[ ] 5. Yes, Fulltime Teacher in Sp. Ed. (Resource)
[ ] 6. Yes, Parttime Teacher in Reg. Ed.
[ ] 7. Yes, Fulltime Teacher in Reg. Ed.
[ ] 8. Other: _____

D. What is your highest degree?
[ ] 1. High school diploma
[ ] 2. Bachelors
[ ] 3. Masters
[ ] 4. Doctorate

E. How much training (in clock hours) have you received in the use of microcomputers in the classroom?
[If you have received NO training, please enter a 0 (zero)]

_____ hour(s)

****

IMPORTANT

IF YOU ARE (OR HAVE BEEN) A TEACHER, PLEASE ANSWER THE FOLLOWING QUESTIONS. IF YOU HAVE NEVER BEEN A TEACHER, PLEASE SKIP THIS SECTION AND PROCEED DIRECTLY TO ATTITUDINAL INFORMATION.

****

F. At what level(s) do you teach?

[ ] 1. Elementary
[ ] 2. Middle school/junior high
[ ] 3. High School
[ ] 4. Middle school/junior high and high school
[ ] 5. All of the above

G. How many years have you taught?

_____ year(s)

H. How many students make up your total case load this year (or the last year you taught)?

_____ student(s)

I. Is there a computer resource person you can go to with questions, problems, ideas? (Check as many as apply)

[ ] 1. Yes, in my school
[ ] 2. Yes, in my school system
[ ] 3. Yes, outside of my school system
[ ] 4. No
[ ] 5. I don't know

J. Do you have access to microcomputers in the following locations?

Yes[ ] No[ ] 1. In my classroom
Yes[ ] No[ ] 2. In the school's computer lab
Yes[ ] No[ ] 3. In the school's office
Yes[ ] No[ ] 4. In my home
Yes[ ] No[ ] 5. Other: _____ _____

K. How many microcomputers do you have access to? (if NONE, please enter a 0)

_____ microcomputer(s)

L. Please indicate whether or not you use a microcomputer for the following instructional purposes:

Yes[ ] No[ ] 1. Remediation in basic skills (reading, math, etc.)
Yes[ ] No[ ] 2. To introduce of new concepts
Yes[ ] No[ ] 3. For curriculum enrichment
Yes[ ] No[ ] 4. For computer literacy/programming training
Yes[ ] No[ ] 5. To reinforce good performance/behavior
Yes[ ] No[ ] 6. For routine drill/practice on curriculum content
Yes[ ] No[ ] 7. For student performance/attendance record-keeping
Yes[ ] No[ ] 8. For IEP development/management
Yes[ ] No[ ] 9. To prepare reports
Yes[ ] No[ ] 10. Other: _____

M. How often do you personally use a microcomputer for any of the purposes that you checked above?

[ ] 1. Daily
[ ] 2. 2-4 times per week
[ ] 3. Once per week
[ ] 4. Once per month
[ ] 5. Less than once per month
[ ] 6. Never

N. Please indicate if you use the following types of computer programs for instructional purposes:

Yes[ ] No[ ] 1. Self-written programs
Yes[ ] No[ ] 2. Commercially available programs
Yes[ ] No[ ] 3. In-house/school system prepared programs
Yes[ ] No[ ] 4. Other: _____

O. Please estimate the average amount of time per day that each of your students used a microcomputer under your direction over the past year (or the last year you taught):
[If NONE of your students used a microcomputer under your direction, please enter a 0 (zero)]

_____ minute(s)

ATTITUDINAL Information

Please indicate your degree of agreement or disagreement with the following statements by circling the appropriate number on the scale provided to the right of each statement.

		Strongly Agree					Strongly Disagree
1.	Having a microcomputer in the classroom is not a distraction to most students.	1	2	3	4	5	6
2.	I am pleased with the variety of programs available for use with special education students.	1	2	3	4	5	6
3.	Errors made by computers could have disastrous effects on our society.	1	2	3	4	5	6
4.	Computer-based instruction motivates students to learn material in which they wouldn't ordinarily be interested.	1	2	3	4	5	6
5.	There is a wide range of effective applications of computer-based instruction in the classroom.	1	2	3	4	5	6

		Strongly Agree					Strongly Disagree
6.	I am confident of my ability to learn to use microcomputers effectively in the classroom.	1	2	3	4	5	6
7.	Using computers in the classroom prepares students to meet the demands of their daily lives in the future.	1	2	3	4	5	6
8.	The use of computers stimulates communication among individuals in our society.	1	2	3	4	5	6
9.	Microcomputers permit me to present instruction in ways I cannot now.	1	2	3	4	5	6
10.	All students should be taught to program computers.	1	2	3	4	5	6
11.	Most of the software programs available are easy for students to use without help from the teacher.	1	2	3	4	5	6
12.	The use of computers enhances the daily lives f individuals in our society.	1	2	3	4	5	6
13.	I am confident I will receive (or did receive) adequate training before and during my initial use of the microcomputer in my classroom.	1	2	3	4	5	6
14.	Computers are generally too complicated for use by the average person.	1	2	3	4	5	6
15.	The educational benefits of computers outweigh the disadvantages of their use in the schools.	1	2	3	4	5	6
16.	Using microcomputers in the classroom enhances students' interpersonal skills.	1	2	3	4	5	6
17.	Most available instructional programs are age-appropriate for use with my students.	1	2	3	4	5	6

18.	I am not worried that someone will steal or vandalize a microcomputer that I am in charge of.	1	2	3	4	5	6
19.	Computer-based instruction has the potential of being more effective than traditional methods.	1	2	3	4	5	6
20.	Computers provide an instructional alternative for students who have difficulty in learning material presented through traditional methods.	1	2	3	4	5	6
21.	I can write microcomputer programs to meet the special needs of my students.	1	2	3	4	5	6
22.	The available instructional programs are adequately sequenced for my students.	1	2	3	4	5	6
23.	The available instructional programs adequately branch students to higher and lower instructional levels when necessary.	1	2	3	4	5	6
24.	Classroom use of microcomputers lessens my instructional duties.	1	2	3	4	5	6
25.	The available instructional programs are appropriate for most of my students' ability levels.	1	2	3	4	5	6
26.	Humans have sufficient control over computers to prevent ill effects of their use in our society.	1	2	3	4	5	6
27.	I am not afraid that I will damage the microcomputer or its programming.	1	2	3	4	5	6
28.	The wide use of computers in our society will create many new jobs.	1	2	3	4	5	6
29.	The average individual can learn to use computers effectively.	1	2	3	4	5	6
30.	I am confident I can choose appropriate microcomputer programs for my students.	1	2	3	4	5	6

Thank you for participating in this survey

Appendix B: Microcomputer Pre-Test

NAME_____

MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION

1. True or False:

____ One byte of memory is a unit of 8 bits in many microcomputers

____ One bit of memory is a unit of 8 bytes in many microcomputers

____ 16K refers to 16,000 bytes of memory

____ 32K refers to 32,000 Kilobits of memory

____ In a microcomputer, information is stored in silicon-based microchips.

2. Memory that is allocated for the temporary storage of programs and data is called: (circle letter)

a. Read Only Memory (ROM)
b. Erasable Memory (EM)
c. Programmable Read Only Memory (PROM)
d. Random Access Memory (RAM)
e. Random Acceptible Memory (RAM)

3. The nonprogrammable memory containing the machine-based instructions is called: (circle letter)

a. Read Only Memory (ROM)
b. Erasable Memory (ER)
c. Programmable Only Memory (POM)
d. Random Access Memory (RAM)
e. Random Acceptible Memory (RAM)

4. Information is permanently stored in many microcomputers on which of the following: (circle letter)

a. "floppy diskettes"
b. hard disks
c. audio cassette tape
d. all of the above
e. none of the above

5. Which of the following is generally not true about caring for diskettes? (circle letter)

   a. they should be dusted frequently
   b. they should never be bent or folded
   c. they should be kept in a dust-free place
   d. they should not be placed on or near a magnetic field (e.g. radios, transformers, metallic surfaces)

6. A diskette which contains the software programs that activate a disk operating system is called a:

   _____

7. The diskette on which data is stored is usually called a:

   _____

8. The slot or peripheral box(es) in which an operator places a diskette for operation of the machine or for storage of data is called the:

   _____

9. "SuperPILOT" is: (circle letter)

   a. programmed in PASCAL
   b. a software program
   c. a lesson authoring system
   d. all of the above
   e. none of the above

10. "Primitive instructions which control the operations of a computer" describes which of the following languages: (circle letter)

    a. FORTRAN
    b. COBOL
    c. MACHINE LANGUAGE
    d. BASIC
    e. LOGO

11. Match the following:

High-level languages	a. COBOL b. FORTRAN c. BASIC
Low-level languages	d. ASSEMBLY e. PASCAL f. MACHINE

12. A device that allows transmission of information through telephone lines between two small computers or between a terminal and a large mainframe computer is called a: (circle letter)

    a. teleprompter
    b. hard disk
    c. modem
    d. telewriter
    e. CRT

13. An arrangement whereby a computer is configured to enable many users to use it at the same time with each person at a separate terminal is called a: (circle letter)

    a. time sharing system
    b. a CRT
    c. a batch mode
    d. none of the above
    e. all of the above

14. Equipment that is external to the computer such as printers, graphics board, cassettes, and disk drives are referred to as: (circle letter)

    a. CRT's
    b. interactive networks
    c. external ports
    d. peripherals
    e. none of the above

15. A software system that controls microcomputer operations is called: (circle letter)

    a. APPLEDOS
    b. TRSDOS
    c. CP/M
    d. all of the above
    e. none of the above

16. If a friend of yours wanted to buy a microcomputer that could "interface" with a number of commercial software programs, you would advise him or her to do which of the following? (circle letter)

    a. find a machine that had a lot of memory storage
    b. consider a machine that had CP/M-directed operations
    c. look at the APPLE's
    d. go to a hard disk

17. The term "external port" refers to: (circle letter)

a. points away from the microprocessor that contain the memory
b. locations on the machine to which peripherals can be attached
c. large computers that receive and transmit data to and from remote sites
d. locations on a machine from which data is unloaded
e. all of the above

18. Most software programs for microcomputers are written in some form of which of the following languages? (circle letter)

a. PASCAL
b. FORTRAN
c. COBOL
d. LOGO
e. BASIC

19. A "mainframe" is a general term referring to: (circle letter)

a. large computers
b. a minicomputer
c. a microcomputer
d. a personal computer
e. none of the above

20. TRSDOS translated means: _____

_____

21. Circle the letter next to the following variables one needs to consider in purchasing a microcomputer:

a. color graphics
b. memory
c. storage equipment available
d. the software that will run on it
e. number of disk drives
f. cost
g. use
h. availability of service
i. type of peripherals to be to be used
j. needs of users
k. manufacturer's reputation
l. service record
m. skill of service personnel

22. "Word Handler" is: (circle letter)

a. a word processing program
b. a software program
c. programmed in BASIC
d. all of the above
e. none of the above

23. "BASIC" is: (circle letter)

a. a set of computer instructions
b. a programming language
c. stored in ROM
d. all of the above
e. none of the above

24. "48K RAM" means: (circle letter)

a. the machine processes information at 48,000 Kilobits a second
b. the machine has a user accessible memory storage bank equal to 48,000 bytes of information
c. the machine can randomly access 48,000 bytes of information per minute
d. all of the above
e. none of the above

25. Typing "DIR" <ENTER> on a TRS microcomputer does which of the following things? (circle letter)

a. gives the user a list of system and user files
b. turns the machine off
c. calls up the word processing information
d. allows the user to store information
e. tells the user how the memory is stored on a diskette

26. To edit the instructions contained in a program written in SuperPILOT, which of the four editors would you enter from the main menu? _____

_____

Appendix C. Microcomputer Post Test

POSTTEST

MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION

1. CPU translated means:

_____

2. The software program that instructs the computer's CPU how to transfer information to and from a disk is called:

_____

3. On a microcomputer with 2 disk drives, the diskette which is inserted into the primary (lower-numbered) drive is usually called a:

_____

4. What is the minimum number of disk drives needed to store a BASIC program from internal memory to a diskette?

_____

5. To edit the instructions contained in a program written in SuperPILOT, which of the four editors would you enter from the main menu?

_____

6. Match the following by placing the correct letter from column A in the appropriate spaces in column B.

A	B
a. Input operations	___Keyboard
b. Process operations	___CPU
c. Output operations	___Printer
d. Both input and output operations	___Disk drive
	___Joystick
	___Monitor
	___Internal memory
	___Voice synthesizer

7. Match the following by placing the correct letter from column A in the appropriate spaces in column B.

A	B
a. High level language	___Word Handler
b. Low level language	___BASIC
c. CAI software	___MACHINE
d. Authoring system	___SuperPILOT
e. Word-processing software	___PASCAL
	___ASSEMBLY
	___Spellmaster

8. True or False:
___One bit of memory is the smallest unit of electronic information in microcomputers.
___One byte of memory is a unit of 8 bits in many microcomputers.
___16K refers to approximately 16,000 bits of memory.
___32K refers to approximately 32,000 bytes of memory.
___In a microcomputer, printed hard copy is used to store information for later machine-based analysis.

****
For questions 9 - 25, circle the letter of the one best answer for each question.
****

9. An operating system that can be used across brands of microcomputers is:

a. APPLEDOS
b. TRSDOS
c. CP/M
d. all of the above
e. none of the above

10. Equipment that is internal to the computer is referred to as:

a. CRT's
b. interactive networks
c. peripherals
d. external ports
e. none of the above

11. "SuperPILOT" is:

a. programmed in PASCAL
b. a software program
c. a lesson authoring system
d. all of the above
e. none of the above

12. An arrangement whereby a computer is configured to serve just one person is called a:

a. time-sharing system
b. a CRT
c. a batch mode
d. a mainframe
e. none of the above

13. Which of the following is generally true about caring for diskettes?

a. they should be dusted frequently
b. they should never be stored vertically
c. they should be cleaned periodically with a mild solvent
d. they should not be placed on or near a magnetic field

14. "Machine language" is:

    a. a set of primitive instructions which control the operations of a computer.
    b. a programming language that closely resembles human language.
    c. a programming language which consists of mneumonics or alphabetic codes.
    d. (b) & (c)
    e. (a), (b), & (c)

15. Most software programs for microcomputers are written in some form of which of the following languages?

    a. PASCAL
    b. FORTRAN
    c. COBOL
    d. LOGO
    e. BASIC

16. A "mini-computer" is a general term referring to:

    a. a large computer
    b. a mid-size computer
    c. a small "desk-top" computer
    d. all of the above
    e. none of the above

17. If a friend of yours wanted to buy a microcomputer for which the most commercial software is available, you would advise him/her to:

    a. find a machine that had a lot of memory storage.
    b. consider a machine that had 2 disk drives.
    c. look at the APPLE's.
    d. go to a hard disk.
    e. all of the above.

18. Locations on the machine to which peripherals can be attached are called:

    a. internal ports
    b. random access devices
    c. external ports
    d. operating systems
    e. none of the above

19. "BASIC" is:

    a. a disk operating system
    b. a high level programming language
    c. stored in an EPROM microchip
    d. (a) & (c)
    e. (b) & (c)

20. "48K RAM" means:

    a. the machine processes information at approximately 48,000 bytes a second.
    b. the machine has a user-accessible memory storage bank equal to approximately 48,000 bytes of information.
    c. the machine can store approximately 48,000 bytes of information on a diskette.
    d. all of the above
    e. none of the above

21. "Word Handler" is:

    a. a word processing program
    b. a software program
    c. a program that only runs on the Apple
    d. all of the above
    e. none of the above

22. Memory that is allocated for the permanent storage of instructions and/or data is called:

    a. Read Only Memory (ROM)
    b. Erasable Memory (EM)
    c. Programmable Reed Only Memory (PROM)
    d. Random Access Memory (RAM)
    e. Random Acceptible Memory (RAM)

23. The temporary portion of memory accessible to the user is called:

    a. Read Only Memory (ROM)
    b. Erasable Memory (EM)
    c. Programmable Only Memory (POM)
    d. Random Access memory (RAM)
    e. Ramdom Acceptible Memory (RAM)

24. When most microcomputers are turned off, information on which of the following will be unaffected?

    a. floppy diskettes
    b. ROM
    c. RAM
    d. (a) & (b)
    e. (a), (b), & (c)

25. Typing "LLIST" <ENTER> when in BASIC on a TRS microcomputer does which of the following things?

    a. gives the user a list of system and user files
    b. lists lines of a BASIC program on a printer
    c. calls up the BASIC editor
    d. allows the user to store information on a diskette
    e. tells the user how the memory is stored on a diskette

Appendix D: Course Evaluation

MICROCOMPUTER APPLICATIONS IN SPECIAL EDUCATION

1984 Course Evaluation

We are very interested in your feelings regarding the usefulness of the workshop/course in which you have just participated. Your feedback will be used in evaluating the course and in revising its content for future presentations.

* *

I. Please rate each module presented in the course on a scale from A (excellent) to F (poor). In addition, please indicate the modules' greatest strengths and greatest weaknesses from your point of view.

MODULE 1: INTRODUCTION TO MICROCOMPUTERS (history of computers, general structure and operations of microcomputer, hands-on experience with ComputerWRITER/DOODLE)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 2: SOFTWARE APPLICATIONS (computer terminology, types of educational software applications, hands-on experience with CITH software--CMMRS, Spellmaster, CRIS, CIRIS)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 3: INTRODUCTION TO WORD PROCESSING (applications of word processing in special education, hands-on experience with Word Handler)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 4: EVALUATING EDUCATIONAL SOFTWARE (general research findings on CBI, evaluation criteria, hands-on experience evaluating educational software)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 5: INTRODUCTION TO AUTHORING SYSTEMS (authoring system applications, hands-on experience with SuperPILOT)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 6: INTRODUCTION TO BASIC PROGRAMMING (discussion of major BASIC commands, hands-on experience with BASIC programming)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 7: LOCAL COMPUTER RESOURCES (Indiana Clearinghouse for Computer Education, visit to local computer store)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

MODULE 8: GENERAL ISSUES (hardware and s tware resources, packaged vs. teacher-developed software, issues and logistical considerations)

Rating (A to F) _____
Greatest strengths:

Greatest weaknesses:

* *

II. Looking back on the course as a whole, please rate the following aspects of instruction on a scale of A (excellent) to F (poor).

	RATING
1. Usefulness of printed handouts	_____
2. Clarity of presentations by instructors	_____
3. Comprehensiveness of presentations	_____
4. Order of module presentation (sequencing)	_____
5. Pacing of instruction	_____
6. Instructors' level of preparedness	_____
7. Helpfulness of instructors	_____
8. Value of "hands-on" experiences	_____
9. Value of field experiences	_____
10. Usefulness of this experience in relation to your present teaching situation	_____

* *

III. Please feel free to use the space below to add any additional comments you may have about any aspect of the institute/course.

* *

1985 Microcomputer Applications in Special Education Course Evaluation

The 1985 Microcomputer Applications in Special Education course utilized the same evaluation instruments as the 1984 course except for the post-test, which was administered via the STEEL/CBTS. Evaluation of the 1985 course consisted of three components: 1) assessment of changes in teacher attitudes toward microcomputers as a result of participating in the course; 2) determination of teacher mastery of the content presented; and 3) assessment of consumer satisfaction with the training experience. Results of these evaluation components are discussed below.

Changes in Teachers' Attitudes

Immediately prior to and following participation in the course, teachers' attitudes toward computers in education were assessed through the use of an attitude scale developed and field-tested by the staff. The attitude scale contains a series of demographic questions followed by 30 statements to which teachers indicate their degree of agreement or disagreement by circling the appropriate number on a 6-point scale (1 indicates strong agreement; 6 indicates strong disagreement.) The demographic questions and list of 30 statements is included as Appendix A of the 1984 course evaluation information.

As can be seen in Table 1, participants' attitudes showed a change of greater than 0.3 points on the 6-point scale on 13 items as a result of participation in the course. Nine of these items showed change of this magnitude on the 1984 attiude scale as well, but three (items 4, 9, and 15) are indicative of improved teacher attitude regarding computer use in the classroom.

- - - - - - - - - - - - - - -

Insert Table 1 about here

- - - - - - - - - - - - - - -

Table 1

Attitude scale statements reflecting a change in teachers' attitudes toward microcomputers as a result of participation in the course.

A. Statements for which a POSITIVE change in attitude occurred

		Mean Ratings (N=20)		
Item #	Statement	Pre	Post	Change
2	I am pleased with the variety with the variety of programs available for use with special education students.	3.50	3.81	+0.31
4	Computer-based instruction motivates students to learn material in which they wouldn't ordinarily be interested.	1.77	2.18	+0.41
9	Microcomputers permit me to present instruction in ways I cannot now.	1.85	2.18	+0.33
11	Most of the software programs available are easy for students to use without help from the teacher.	3.33	3.90	+0.57
13	I am confident that I received adequate training before and during my initial use of the microcomputer in my classroom.	2.50	3.00	+0.50

15	The educational benefits of computers outweigh the disadvantages of their use in the schools.	1.54	2.18	+0.64
18	I am not worried that someone will steal or vandalize a microcomputer that I am in charge of.	3.00	2.18	-0.82
23	The available instructional programs adequately branch students to higher and lower instructional levels when necessary.	3.27	4.00	+0.73
27	I am not afraid that I will damage the microcomputer or its programming.	2.23	1.72	-0.51
28	The wide use of computers in our society will create many new jobs.	2.23	3.63	+1.40

B. Statements for which a NEGATIVE change in attitude occurred.

		Mean Ratings (N=20)		
Item #	Statement	Pre	Post	Change
8	The use of computers stimulates communication among individuals in our society.	2.85	2.54	-0.31

21	I can write microcomputer programs to meet the special needs of my students.	4.45	3.36	-1.09
25	Humans have sufficient control over computers to prevent ill effects on their use in our society.	2.67	2.09	-0.5

Teachers' responses indicated a positive change in attitude toward ten of the attitude scale statements. The majority of these statements (items 2, 4, 9, 15, and 23) dealt with the teachers' perceived usefulness of the computer as part of their educational curriculum. Teachers became more positive in their attitudes toward using computers and available computer software with their students. It should be mentioned that the software used for evaluation purposes was different than that used the previous year, so the teachers' indication of appropriate branching and ease of use (items 11 and 23) was more favorable than the previous years' ratings.

Responses to three of the statements indicated a negative change in teachers' attitudes as a result of the course. All of these were different than those rated negatively the previous year. One of the strongest feelings expressed by course participants was the perceived difficulty of programming with the BASIC language. In fact, this item (number 21) received the greatest change of any attitude scale item rated, with a mean loss of 1.09 points. Participants clearly felt that the short module on programming of student software was unable to meet their present needs. Taking these strong feelings into consideration, project staff decided to offer a single course devoted to writing software programs directly applicable to classroom students. In the time prior to the next course offering, the module dealing with authoring languages was revised and augmented with additional information/material, and the following summer a course was offerred for teachers who had already taken introductory microcomputing courses. It was designed to extend their skills and to produce useable products for students in the public school classrooms. The course evaluation for this revised course is included in this report.

Mastery of Course Content

Pre-and posttest versions of an information mastery test were administered prior to and following the course, respectively. Copies of these tests are included in the 1984 course evaluation. The major difference of the 1985 course was that, after the pre-test was administered using paper-and-pencil format, the items were entered into the STEEL/CBTS using DAL. Student scores on the pre-test ranged from 7 to 38 points out of a total of 47 points, with a mean score of 22.28.

Post-test items, as previously mentioned, were administered using the CBTS. Test items identical to those used on the previous years' pre- and post-tests were used, and consisted of true-false, multiple choice, matching, and short answer format. Students were required to pass the Computer Literacy test (mastery level was set at .90, non-mastery at .60) in order to successfully complete the course. Students had to log-on to the Cyber 855, and were allowed to take the test as many times as needed to obtain the prespecified mastery level. If, for instance, a student failed to reach criterion on their first attempt, they were able to note the items they had missed as use these as a basis for further study before attempting the test a second time. Using this system, 4 students out of 14 (one student had to drop out of the course due to illness) passed the test on the first attempt, and the number of attempts required to attain mastery ranged from 1 to 4, with a mean of 2.5.

The students had many comments regarding the CBTS, ranging from highly favorable to dislike. Although no data was systematically obtained, it was the observation of project staff that the individuals more familiar with computers were those with more favorable attitudes regarding this type of testing situation. The only other frequently mentioned comment relaced to the inability to visually examine items answered previously. Some students

felt this strategy helped their performance level on tests, and missed having the ability to review items. These suggestions were all relayed to the CBTS developer.

Consumer Satisfaction With the Course

Finally, immediately following the course, participants were asked to provide feedback regarding the usefulness of the experience. This feedback was collected through the use of the evaluation form which is included in the appendices for the 1984 course evaluation.

The evaluation form asked participants to : 1) rate each of the eight modules on a scale from A (excellent) to F (poor) and to indicate each module's greatest strengths and weaknesses; 2) rate ten aspects of the course as a whole on the scale from A to F; and 3) provide additional comments which might aid in evaluation and/or revision of the course.

Participants' ratings of the eight modules and the course as a whole are displayed in Table 2.

- - - - - - - - - - - - - - - - -

Insert Table 2 about here

- - - - - - - - - - - - - - - - -

The modules on software applications, word-processing, and general issues regarding microcomputers received the highest ratings of 3.7[illegible], 4.00, and 3.83 respectively (on a 4.00 scale). The module on BASIC programming received the lowest rating, 3.00, primarily because participants indicated more time was needed to absorb instruction and practice the programming techniques.

As also shown in Table 2, all ten aspects of the course evaluated received a mean rating of above 3.50 on the four-point scale. Preparedness and helpfulness of the instructors, value of course experiences, and usefulness of the experience to present teaching situations were rated most

Table 2

Participants' ratings of the eight modules and the course as a whole

MODULE RATING	Mean Rating *
1. Introduction to Microcomputers	3.33
2. Software Applications	3.75
3. Introduction to Word Processing	4.00
4. Evaluating Educational Software	3.08
5. Introduction to Authoring Systems	3.58
6. Introduction to BASIC Programming	3.00
7. Local Computer Resources	na
8. General Issues	3.83

GENERAL RATINGS	Mean Rating *
1. Usefulness of printed handouts	3.92
2. Clarity of presentations by instructors	3.58
3. Comprehensiveness of presentations	3.75
4. Order of module presentation (sequencing)	3.92
5. Pacing of instruction	3.58
6. Instructors' level of preparedness	4.00
7. Helpfulness of instructors	4.00
8. Value of "hands-on" experiences	4.00
9. Value of field experiences	4.00
10. Usefulness of this experience in relation to your present teaching situation	4.00

* 4.00 = A(excellent); 0.00 = F(poor).

highly at 4.00. The lowest rating was given to clarity of presentation by instructors and pacing of instruction, at 3.58 each. Overall, the participants expressed a high level of satisfaction with the course as a whole and the instructors' helpfulness. Comments substantiated these ratings. Some representative comments include: "I thoroughly enjoyed this course, and am sorry that time did not allow for further instruction," "Quite a team! Fast, furious, and frustrating summer made less frenzied by the human touch inherent to all of your approaches," "Really well done--I can't tell you how much more comfortable I am with it all now. Feel I can build on this now," "When I wasn't understanding a new concept, I knew not to panic. I knew the instructors would teach me until I had it even if it took until December. The course has been invaluable! There was no pressure, only a supportive learning atmosphere. You all practice what you preach!", and "I enjoyed this class tremendously. The information presented really helped me understand more about computers and I feel I will use my classroom computer more effectively next year."

1986 SuperPILOT for Teachers Course Evaluation

Description

The Center for Innovation in Teaching the Handicapped at Indiana University offered a summer workshop for teachers to develop software programs for their students using the SuperPILOT lesson authoring language. The workshop was held on the Bloomington campus from June 2 - June 18, 1986. Twenty teachers attended the workshop, and each developed an individually-designed software program and completed a criterion referenced test demonstrating their mastery of SuperPILOT commands. A description of the course and evaluation by the participants is included in this report.

In the majority of schools, teachers must rely on commercial software programs which may not fit their specific needs or complement their existing educational goals. The summer workshop was designed to provide teachers with specific skills which would enable them to conceptualize, design and create software programs which would supplement and/or extend their existing curriculum. It also allowed teachers who had attended previous microcomputer courses the opportunity to extend the skills in using authoring languages.

Information advertising the course was sent to school systems throughout south central Indiana in early May. Teachers were instructed to either mail their application or call a contact person at the Center for Innovation in Teaching the Handicapped (CITH). Applications were accepted until May 28, and then were screened by CITH staff members for inclusion in the course.

Participant Selection

Initial screening criteria included: (1) previous experience using microcomputers; (2) availability of an Apple microcomputer; (3) access to microcomputers in the school setting. These criteria were selected because this course was not designed to be an introductory microcomputing course. Rather, it was aimed at teachers who had already completed basic courses, either with CITH, through the university, or those offered by the regional Computer Information Centers. Additionally, it was felt that there would be a need for participants to devote a number of hours outside of class time for program development. Since many of the participants commuted. availability of a microcomputer in their home or school was deemed important. Finally, it was considered important that there be an Apple microcomputer in the teachers' classrooms so that their students could use the teacher-created program and provide feedback regarding its effectiveness, who could then modify it if necessary.

Due to a lack of computing facilities available for the SuperPILOT course on the Bloomington campus course enrollment was limited to twenty teachers. The instructors felt that the amount of time each participant acutally spent on the microcomputer during class time was important, and so quality of instruction was emphasized over quantity of service provided. Teachers attending the workshop came from throughout south central Indiana, with teachers from Carmel, New Castle, Mooresville, Nashville, Washington, Linton, and Bloomington represented. The schools in which they taught ranged from rural to suburban, and from elementary to high school. Teachers were divided almost equally between special and regular educators, and there was also a wide range of ages represented.

Course Objectives

Because of the relatively short amount of time, the content of the course was carefully restricted to the most important concepts needed by teachers to develop functional programs. The major objectives included:

1. Participants will be able to utilize the major steps in the Computer Assisted Instruction (CAI) lesson development process.
2. Participants will demonstrate awareness of considerations in the development of instructional software by discussing specific ideas and their feasibility with the instructors and other class members.
3. Participants will be able to describe the major functions of the SuperPILOT authoring system and will be able to use this knowledge to write computer-based material for their students.
4. Participants will demonstrate the use of various SuperPILOT commands.
5. Participants will create and edit files within the SuperPILOT Lesson Text Editor.
6. Participants will create and edit graphics and/or sound files within the SuperPILOT Graphics and Sound Editors, respectively.
7. Participants will produce one lesson for use with their students.
8. Participants will successfully pass a criterion-referenced test covering SuperPILOT commands with 80% accuracy.

Logistical Considerations

Teachers participating in the workshop had access to Apple microcomputers in their homes or schools. The instructors felt that individual practice in programming was essential for ensuring mastery of the SuperPILOT language, so twenty copies fo the authoring system were purchased for the course. These were assigned to individual teachers to take home for work outside of class time. A copy of the software loan agreement is included in Appendix A.

The course was taught in two different locations--the CITH classroom and the Apple Lab in the School of Education. The CITH classroom was used to present CAI/lesson development instruction and some SuperPILOT instruction. It was also used for "show-and-tell" sessions. For work on individual projects, the class moved to the School of Education computer classroom. This room had 13 Apple computers, but only 10 or 11 were available for class use at any one time. This created a situation where two teachers would work at a computer as a dyad, helping to provide suggestions and debug each others' programs. The instructors circulated when the class met in the computer classroom, helping students who indicated a need for assistance.

Course Content

Due to the limited amount of time available in which to teach the course, it was necessary to carefully define the content to be covered. Prior to the first meeting, a series of planning sessions were conducted and the course schedule/syllabus constructed. Copies of these documents are included in Appendix B. Topics and activities were listed and required readings and homework assignment specified. A copy of the

syllabus and schedule was distributed to each participant on the first day of class. The schedule was followed throughout the course, and all noted topics were covered.

Evaluation

At the conclusion of the course, participants were required to take a criterion-referenced test covering SuperPILOT commands. An initial mastery level of 90% was specified. Scores ranged from 68% to 100%. However, the person achieving the perfect score had had previous experience with the PILOT authoring language, so the next highest score (96%) was considered to be the "high" score. The mean score of the entire class was 87% (SD 9.14). The instructors felt that the difficulty level of the test required mastery level to be lowered to 80%. At this level, four participants failed to reach the specified criterion. These individuals were contacted and asked to return to Bloomington to discuss/review their mistakes, study the material and retake the test when they felt re Each did so and reached criterion in the second testing situatio

Participants also were asked to evaluate the course and the instructors. This was done using a form containing a number of statements related to the course and having teachers rate each statement on a scale from 1 (Strongly Disagree) to 5 (Strongly Agree). A copy of the statements and their mean scores and standard deviations is included in Appendix C.

The ratings indicate a strong perception that the course was worthwhile and that the participants will use the information learned when they return to their classrooms. Furthermore, since many of the teachers were working at the same level, e.g., elementary, middle or high school, there was a great deal of program exchange of the final day

of the course. The major criticisms involved logistical considerations, such as the number of available microcomputers on class days and the required purchase of the CAI lesson development textbook. Overall, the course was judged to be valuable and of future benefit to all participants, as well as to their students.

Appendix A: Software Loan Agreement

Center for Innovation in Teaching the Handicapped
Indiana University

***** SOFTWARE LOAN AGREEMENT *****

In borrowing the microcomputer software specified below, I acknowledge that I have read the provisions for its care and use and agree to abide by those provisions. I also verify that I understand the stated consequences for the violation of these provisions.

-SuperPILOT Editor's Manual
-SuperPILOT Language Reference Manual
-SuperPILOT Author Disk
-SuperPILOT Lessons Disk
-SuperPILOT Log Disk

Copy Number:_____

Loan Provisions

1. This program is copyrighted and copy protected. I will not copy this program nor attempt to copy it.

2. I will not write in either of the manuals, nor will I remove the reference cards in the back of each manual.

3. I will exercise due care in the use of this program and will ensure that the program disk is not damaged in any way. I will operate the Author Disk at all times with a write-protect tab correctly in place.

Consequences

1. I understand that I am liable for prosecution under the federal copyright law if I copy the above software.

2. If I lose this software package, or the Author Disk only, I must pay $125.00, which is the cost of replacing the program.

3. If I lose either of the manuals, or if I write in either of the manuals, I will pay $25.00 for each one that must be replaced.

4. I will not be held liable for normal wear and tear on the manuals or the storage box for the program.

Any fees owed will be deducted from the $300.00 stipend. I have read the above conditions, understand them, and agree to abide by them. I have inspected the program materials and have found them to be in good condition with the following exceptions:

Signature	Date Borrowed	Date Due

Appendix B: Schedule/Syllabus

Schedule: SuperPILOT for Teachers
M-W-F 11:30 a.m. - 4:30 p.m.

Date	Location	Topic	Estimated Times
6/2 (M)	CITH	Introduction: Syllabus Who's Who Credit Registration Software Checkout Textbook	11:30-12:15
		Basics of microcomputer operation	12:15-12:30
		LUNCH	12:30-1:00
		Discussion: Types of CAI	1:00-1:30
		Discussion: Evaluating CAI	1:30-2:00
		Software Critiques	2:00-2:30
		Discussion: The instructional development process	2:30-3:30
		Begin working on steps #1-4 for individual projects	3:30-4:30
		ASSIGNMENT: -Complete steps #1-4 for individual projects. (p. 311) -Read Chapter 14 in Alessi and Trollip text. -Review Chapters 12 and 13 in Alessi and Trollip text. (optional) -Read Jay article.	
6/4 (W)	CITH	Discuss homework assignments. 2-min. presentation for each	11:30-12:30
		LUNCH	12:30-1:00
		Lecture/Demonstration: Storyboarding	1:00-2:00
		Work on individual storyboards for project	2:00-4:30
		ASSIGNMENT: -Complete storyboard for project.	

6/6 (F)	CITH	Review of storyboard homework Discuss problems, modifications	11:30-12:00
		Overview of SuperPILOT	12:00-12:15
		Initialize lesson diskettes	12:15-12:30
		LUNCH	12:30-1:00
	Apple Lab	Intro to Editor commands	1:00-2:00
		SuperPILOT commands: T, A, M, J, U	
		Structure of SuperPILOT lesson	
		Text modification commands - TS	2:00-2:30
		Intro to variables	2:30-3:00
		Creation of pause routine	3:00-3:30
		Begin programming lesson	3:30-4:30
		ASSIGNMENT: -Read SuperPILOT command summary handout. -Begin reading SuperPILOT language manual. -Program title screen and first page of lesson.	
6/9 (M)	CITH	Small groups - Show first 2 screens of lesson	11:30-12:30
		Discuss problems, modifications	
		LUNCH	12:30-1:00
	Apple Lab	Review of commands from 6/6	1:00-1:15
		Remark Statements	1:15-1:30
		Modifying viewport	1:30-1:45
		Use of conditioners	1:45-2:30
		Use of variables including system	2:30-3:15
		Work on programming lesson	3:15-4:30
		ASSIGNMENT: -Continue reading SuperPILOT language manual. -Continue programming lesson.	

6/11 (W)	CITH	Small groups - Show lessons to date	11:30-12:00
		Intro to graphics files	12:00-12:30
		LUNCH	12:30-1:00
	Apple Lab	Graphics commands	1:00-1:45
		Play with graphics	1:45-2:30
		Keeping score during lesson	2:30-3:00
		Work on programming lesson	3:00-4:30
		ASSIGNMENT: -Continue reading SuperPILOT language manual. -Continue programming lesson.	
6/13 (F)	CITH	Small groups - Show lessons to date	11:30-12:00
		Intro to sound	12:00-12:30
		LUNCH	12:30-1:00
	Apple Lab	Create sound routine	1:00-1:45
		Linking lessons	1:45-2:15
		Use of animation	2:15-3:15
		Overview of recordkeeping	3:15-3:30
		Work on programming lesson	3:30-4:30
		ASSIGNMENT: -Continue reading SuperPILOT language manual. -Continue programming lesson.	

6/16 (M)	Apple Lab	Work on lessons	11:30-1:00
		LUNCH	1:00-1:30
	CITH	Review for test, questions, problems	1:30-2:30
6/18 (W)	CITH	SuperPILOT command test	11:30-12:15
		Course Evaluation	12:15-12:30
		LUNCH	12:30-1:00
		Show-and-Tell	1:00-4:30

SYLLABUS: SuperPILOT for Teachers

Instructors:
Chris Bahr 335-0172
Sara Hindman 335-9753
Lew Polsgrove 335-9775

CITH Lab hours: Tuesday and Thursday, 9 am - 4 pm

Apple Lab hours: 9:30am - 9pm Mon.-Thurs.; 9:30am - 5pm Fri.; 10:30am - 4:30pm Sat.; 1pm - 5pm Sun.

Course Description: This course is designed to provide teachers with basic lesson design considerations and SuperPILOT commands which will allow them to create lessons for use in their classrooms.

Course Objectives:

1. Participants will be able to utilize the major steps in the CAI lesson development process.
2. Participants will be able to describe the major functions of the SuperPILOT authoring system and use this knowledge to write computer-based material for their students.
3. Participants will demonstrate the various SuperPILOT commands.
4. Participants will describe the names and functions of the four editors.
5. Participants will create and edit files within the SuperPILOT Lesson Text Editor.
6. Participants will create and edit graphics within the SuperPILOT Graphics Editor and/or sound within the SuperPILOT Sound Editor.
7. Participants will produce one lesson for u with their students.
8. Participants will successfully pass a criterion-referenced test covering SuperPILOT commands and capabilities.
9. Participants will leave the institute with basic SuperPILOT authoring skills and one individually relevant SuperPILOT program.

Evaluation :

1. One SuperPILOT lesson completed to course criteria.
2. Criterion-referenced test on SuperPILOT commands at end of course.

Text : Alessi, S. M., & Trollip, S. R. (1985). Computer-Based Instruction: Methods and Development, Prentice-Hall, Inc.

Appendix C: Course Evaluation

COURSE EVALUATION: K500 SuperPILOT for Teachers
(1986, n=19)

	MEAN	SD
1. I learned more in this course than I have in most others.	3.94	0.73
2. The course was kept on schedule.	4.53	1.10
3. This course was well organized.	3.89	1.10
4. The instructors were well prepared for class meetings.	4.58	0.61
5. The instructors were able to explain the subject clearly.	4.42	0.61
6. The instructors made difficult topics easily understandable.	4.11	0.66
7. The instructors used examples and illustrations well.	4.58	0.61
8. The instructors actively helped students with problems.	4.79	0.54
9. The instructors respected student questions about the subject.	4.84	0.50
10. The instructors used teaching methods well suited to the course.	4.47	0.61
11. The instructors recognized when students failed to comprehend.	4.16	0.60
12. The instructors adapted to student abilities, needs and interests.	4.28	0.89
13. The instructors provided sufficient help.	4.37	0.83
14. The instructors made me feel free to ask questions in class.	4.74	0.45
15. Daily class activities were clearly tied to course objectives.	4.68	0.58
16. The content of this course is worthwhile.	4.63	0.49
17. Assignments were of definite instructional value.	4.32	0.67

18. Developing the course project was a good learning experience.	4.74	0.45
19. I am pleased with the text required for this course.	2.67	1.19
20. Duplicated handouts were valuable supplements for this course.	4.68	0.58
21. The amount of material covered in the course was reasonable.	3.95	1.03
22. The amount of work required in this course was adequate.	4.17	0.86
23. SuperPILOT is a useful programming language for teachers to know.	4.58	0.61
24. The information I learned in this class will help me create relevant software programs for my pupils.	4.58	0.51
25. This course motivates me to do further independent programming.	4.26	0.81
26. I plan to continue using SuperPILOT to develop software.	3.94	1.10
27. This course will be of practical benefit to me.	4.26	0.81
28. My experience in this class will help me to evaluate commercial software for my students.	4.22	0.73
29. I learned a lot in this course.	4.47	0.51
30. I developed the ability to solve actual problems in this course.	4.00	0.94
31. Team teaching provided insights a single instructor could not.	4.42	0.61
32. Instruction was well coordinated among the team teachers.	4.47	0.51
33. I liked the team teaching approach in this course.	4.68	0 48