DOCUMENT RESUME

ED 255 192 IR 011 562

TITLE	Computers in Education: Goals and Content.
INSTITUTION	California St- Dept. of Education, Sacramento.
PUB DATE	85
NOTE	22p.
AVAILABLE FROM	Bureau of Public tions, California State Department of Education, PO Box 271, Sacramento, CA 95802-0271 ($2.50 per copy plus sales tax).
PUB TYPE	Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	*Computer Assisted Instruction; *Computer Literacy; *Computer Science Education; Computer Software; *Educational Objectives; Learning Activities; *Problem Solving; Program Implementation; Skill Development; *Social Problems
IDENTIFIERS	California; *Computer Uses in Education

ABSTRACT

This document offers a perspective on the use of computers in education to assist California educators with the design and implementation of educational activities that encourage the effective and creative use of computers and computer software throughout the school curriculum. It is organized in six strands, each identifying a basic core of skills. Strands cover the following topic areas: (1) operation of computer systems; (2) computer assisted learning; (3) computer applications, including word processing, electronic spreadsheets, database managers, and programming; (4) problem solving with computers, including communication skills and the thinking skills of describing the problem, breaking down the problem, developing algorithms, anticipating outcomes and finding unforseen results, generalizing a problem solution, and testing and debugging; (5) computer science; and (6) societal impact of computers. A rationale for each strand explains important uses of or presents general knowledge about computers. Examples of learning activities are included to clarify the intent of the content, but grade levels for their use are not indicated. (LMM)

ill H nig—Superintendent of ction Sacramento, 1985

Publishing Information

Computers in Education: Goals and Content was prepared under the direction of the California State Board of Education and approved unanimously by that body on July 13, 1984. The members of the Board were Sandra Boese, President; Henry Alder; Josie Gray Bain; Agnes Chan; Daniel M. Chernow; Gloria Hom; Angie Papadakis; Kenneth Peters; David T. Romero; Mark Sedway; and John Ward. This publication fulfills a requirement in Senate Bill 813 (Chapter 324, Statutes of 1983). Section 226 of SB 813 states:

> The State Board of Education, upon recommendation by the Superintendent of Public Instruction, shall approve a model course of study in computer education for pupils in kindergarten and grades one through twelve and shall submit the course of study to the Legislature for review on or before July 1, 1984.

Computers in Education, which was prepared for the Board of Education and the Superintendent of Public Instruction by the Educational Technology Unit, was edited and prepared for photo-offset production by the Bureau of Publications, was published by the California State Department of Education, 721 Capitol Mall, Sacramento, CA 95814-4785, and was distributed under the provisions of the Library Distribution Act and *Government Code* Section 11096.

Additional copies of this document are available for $2.50 each, plus sales tax for California residents, from the Bureau of Publications, California State Department of Education, P.O. Box 271, Sacramento, CA 95802-0271.

See page 17 for information about catalogs and publications available from the Department of Education.

Contents

BEST COPY AVAILABLE

Acknowledgments

THIS DOCUMENT was prepared with the help of a group of educators, each of whom has had extensive experience in using computers in educational settings. As a group they also have had classroom teaching experience, teacher training experience, and planning experience at the school district and county levels. The State Department of Education is grateful for their contributions.

Principal Writer
Martin L. Harris
Computer Access
Davis, California

Project Coordinator
Wendy Harris
Manager, Educational Technology Unit
State Department of Education

Committee Members

Harry Bakker
Director, Instructional Materials Display Center
Office of the Stanislaus County Superintendent of Schools

Jennifer Better
Formerly: Curriculum Coordinator, Cupertino Union Elementary School District
Currently: Developmental Engineer, Hewlett-Packard Company

Jane Gawronski
Assistant Superintendent for Educational Programs and Services
Walnut Valley Unified School District

Bobby Goodson
Formerly: President, Computer-Using Educators, Inc., and Computer Resource Teacher, Cupertino Union Elementary School District
Currently: President, International Council for Computers in Education, and Consultant on Computers in Education

Helen Joseph
Formerly: Computer Resource Teacher, Novato Unified School District
Currently: Computer Education S[illegible]ialist, EQUALS Project, Lawrence Hall of Sc[illegible], University of California, Berkeley

James McCauley
Educational Computing Specialist, Office of the Santa Clara County Superintendent of Schools

Judy Powers
Educational Computing Specialist, Office of the Santa Clara County Superintendent of Schools

Martha Ramirez
President, Computer Literacy, Berkeley, California

Diane Kramer Siri
Formerly: Director, Institute of Computer Technology, Sunnyvale, California
Currently: Assistant Superintendent, Educational Services, Office of the Santa Cruz County Superintendent of Schools

Introduction

THE PURPOSE of this document is to provide a perspective on the use of computers in education and to assist California educators with the design and implementation of educational activities that encourage the effective and creative use of computers and computer software throughout the school curriculum.

The document is organized in strands, each of which contains content that can be used to enhance the educational program by teachers who are using computers in any area of the curriculum. Indeed, the potential uses of the computer are far too diverse and exciting to propose that a computer education course of study be implemented in isolation from other subjects.

Each of the six strands presented below presumes that all students should have the opportunity to develop at least a basic core of skills needed to use computers for two important reasons:

- The computer can be used as a powerful tool to enhance learning.
- The computer will have a pervasive impact on our information-based society and thus on the life and career of every student.

▶ Schooling and the Information Age: Computer Education Goals

This document makes the fundamental assumption that the computer should be viewed as an essential educational tool, one which can enhance student learning at all levels and which can be used to expand the scope of the curriculum in many innovative ways. Thus, all students, regardless of their socioeconomic status, ethnicity, gender, or general level of academic achievement, can benefit from, and should be exposed to, learning experiences with computers. This assumption is made because a rapidly growing number of teachers and administrators with valuable assistance from individuals in the computer industry, have demonstrated that computers (and the expanding array of computer-based peripheral equipment and educational software) can be used in effective ways to enrich the school curriculum and to promote student excitement and success with learning in preparation for life and careers.

This assumption also serves as the basis for the following goals, which should be part of every effort to incorporate computers into the schools' curriculum:

- Computers should be integrated as learning tools throughout the curriculum in each subject area consistent with the overall educational goals of the school and district.
- Computers should be utilized in programs that prepare students for jobs and occupations that are either affected by or centered on the use of computers.
- Extensive hands-on computer experience should be provided for all students at all grade levels.
- The most effective learning experiences will be those structured so that the student has control of the computer and the way in which information is handled; the computer should not be used primarily as an electronic text which provides information to the student.
- At an informal level all students should acquire some fundamental programming skills.
- Teachers at every level and in all subject areas should recognize and take advantage of the computer as a powerful and concrete device for teaching critical-thinking and problem-solving skills.

▶ The Information Age and Computers

A second major assumption of this document is that all students must be prepared to utilize computers in order to take full advantage of the rapid technological changes occurring in the workplace and the home. It is apparent that our society is moving fast into the information age. The computer will be an increasingly dominant tool as this transformation takes place, just as other tools, such as the wheel and the printing press, were vital to fundamental changes in the past.

The manner in which information is processed in business, industry, and agriculture has been revolutionized by the computer, making the occupational and career implications of computers an important area to be addressed in curriculum development. At the consumer level an increasing number of homes already have information centers; our means of transportation are increasingly controlled by computer systems; and most of our basic material needs (shopping, communications, medical care, financial transactions, and so on) are already undertaken with the assistance of computers. Future applications likely will have an even more profound impact on all of us.

Curricular Strands

This document is organized into six major strands:

1. Operation of Computer Systems
2. Computer-Assisted Learning (CAL)
3. Computer Applications
4. Problem Solving with Computers
5. Computer Science
6. Societal Impact of Computers

While students will develop different levels of expertise, prior to graduation all students should be expected to master a basic core of skills in using computers. Thus, each strand identifies a basic core of skills. In addition, a few strands identify skills for more motivated or proficient students who would benefit from the acquisition of more advanced skills and concepts.

Each strand has a rationale that explains important uses of computers or presents general knowledge about computers. Examples of learning activities are included in most strands to clarify the intent of the content. Since students will be ready to learn particular skills and benefit from a variety of activities at different times, these examples are not suggested for use at a particular grade level.

These strands should be presented in a parallel fashion, so that a given use of computers will address skills and knowledge from several strands. For example, learning a programming language in third grade involves skills presented in the operation of the computer systems strand, the problem solving with computers strand, and the computer applications strand.

Finally, while the strands give emphasis to the use of the computer as a learning tool, the computer will also be an object of instruction; that is, attention will be given to the computer, its history, how it works, how it can be used to perform useful tasks, and a discussion of the need to protect against misuses of the computer. The potential career implications of the computer also need to be considered.

Integration of Computer Education

As previously stated, the purpose of this document is to assist educators in designing computer activities that are integrated across the curriculum. In developing curriculum plans, educators must recognize the need for flexibility, particularly as understanding grows about how best to use computers and what types of software are most appropriate and effective in the learning process.

Computer-education curriculum plans should recognize that the needs of students at different levels will change as current programs begin to have an impact. For example, in a number of districts subject matter considered suitable only at the high school level five years ago is now introduced at the junior high school level and likely will be used routinely in the elementary grades within the next few years. In other districts current limitations in equipment and software may require schools to focus initial services on a particular grade level or a particular curricular area. In such situations, as additional equipment is acquired, expansion into other subjects and to additional students should be managed in planned stages.

In addition, the potential uses of computers are expanding almost daily because of new developments in hardware and a growing array of well-designed software. With training and experience, teachers are developing increasingly sophisticated uses for existing applications software (word processors, programming languages, research and analysis tools such as spreadsheets and data-base managers, lesson-authoring systems, and student-information management systems). Thus, the skills to be acquired by students and the teaching strategies and techniques used by teachers that take advantage of these rich new resources will have to be reassessed regularly by each district and school.

This document does not specifically define *computer literacy*. Rather, by laying out a broad array of skills and knowledge, the document covers all the topics that could be considered under the rubric of computer literacy. Anyone wishing to design a computer literacy program may find all the relevant information here. Those wishing to design educational uses of the computer without using the term *computer literacy* can also find the relevant content in this document.

An ambitious and difficult agenda for California schools is presented, one that is designed to remain relevant for a number of years. Implementing the ideas contained here will be a long and sometimes difficult effort and will require substantial curriculum planning and development. However, these developmental efforts should not focus solely on computers. They must reflect the educational goals and priorities of each district and school, other ongoing curriculum reform efforts, and what is known about effective teaching and learning. Computers should be used to further the overall educational purpose of a school, not be adopted merely as an "add on" project.

Implementation plans should reflect the educational goals and expected learner outcomes specified in any master plan a district may develop. At the classroom level specific unit objectives that incorporate computers in the curriculum should also reflect district goals.

▶ *Implementation Issues*

The scope of this document does not permit a discussion of the critical issues of hardware and software selection; nor does it consider the strengths and weaknesses of the various options for utilizing hardware (such as a laboratory versus placing computers in each classroom). A variety of other documents, including a number of computer education journals, provide this type of information. The curriculum development efforts described above, though important, will not be successful without a commensurate effort in the training of staff to use computers in the school effectively.

Operation of Computer Systems

THE OPERATION of the computer systems strand deals with skills and knowledge students require in order to operate a computer system successfully and safely. It is important that the content of this strand be taught in a hands-on fashion in conjunction with the study of material from other strands. In this manner young students should be learning to identify and use alphabetic and numeric keys in the context of using instructional software or a simple programming language. These experiences should begin as early as possible in a student's educational career, preferably in the primary grades.

This strand is about computer hardware, software, and functions. While the specific ways of using a computer vary from system to system, students should learn that the basic functions listed here are available on any system. Perhaps this can be best accomplished by ensuring that students have the opportunity to work with a wide range of hardware and software during their educational careers. The computer systems strand includes:

1. Keyboarding Skills
   a. Identify and use alphabetic and numeric keys.
   b. Identify and use common special-purpose keys such as RESET, BREAK, ESC, RETURN, ENTER, RUBOUT, DELETE.
   c. Identify and use function keys appropriately.
   d. Identify and use numeric keypads.
   e. Develop keyboarding skills at the touch control level (touch typing).

2. System Components
   a. Describe the purpose of the major components of a computer system and their interrelationships:
      (1) The computer: random access memory (RAM), read-only memory (ROM), and central processing unit (CPU), mother board, slots, integrated circuits, and circuit boards.
      (2) Input/output devices: keyboard, monitor or TV, light pen, graphics tablet, voice synthesizer, modem, music synthesizer, matrix printer, letter-quality printer, joysticks, paddles, robots
      (3) Storage devices: cassette, floppy disk, hard disk, video disk
      (4) Local- and wide-area networks and telecommunications systems

3. Computer Operations
   a. Demonstrate the proper care of software and hardware.
   b. Perform the following tasks with software: load, list, erase, copy, stop, break, continue.
   c. Respond appropriately to system error messages.

4. Computer Applications
   Computer applications are presented in a substantive manner in the next strand. The applications listed here may be introduced in a general way before actual hands-on experience is provided:
   a. Describe the characteristics and uses of a variety of procedural programming languages.
   b. Describe the characteristics and uses of applications programs and nonprocedural languages such as:
      (1) Spreadsheets
      (2) Database and file managers
      (3) Word processors
      (4) Bulletin boards
      (5) Electronic mail
      (6) Simulations and games
      (7) Music synthesizers
      (8) Graphics generation programs

Computer-Assisted Learning

COMPUTER-ASSISTED learning (CAL) is the most common computer application in schools today, and CAL materials are frequently placed in one of three broad categories: drill and practice, tutorials, and simulations. While these categories are not mutually exclusive, they represent an increasing level of student control over the computer and the educational material being presented. They also represent increasing opportunities for students to learn and apply critical-thinking and problem-solving skills. Ideally, all students should have experience with all types of CAL, where appropriate, throughout the curriculum:

1. Drill and practice programs are designed to reinforce and/or remediate previously learned skills and knowledge. They are electronic versions of work sheets that can provide immediate feedback to the student and provide harder or easier levels of drill in response to student performance. Many drill and practice programs also provide a student management component that keeps track of student performance and reports it to the teacher.
2. Tutorials are designed to teach new material in a manner similar to a teacher working one-on-one, usually through a Socratic-like dialogue.
3. Simulations are computer-based models of real-life events that allow the student control over certain variables and provide ways of testing hypotheses about those variables. They provide a method of conducting experiments, such as those in chemistry, without the need for a laboratory, equipment, and chemicals. Other simulations allow students to become involved in historical events or to simulate actual business operations.

Another frequently described category of CAL is educational games. Rather than being a distinct category, however, educational games are more of a presentation strategy. Any of the three types of CAL described above can be in a game format to add motivation for its completion.

However well designed computer-assisted-instruction software may be, and whatever level of control it appears to provide the student over the educational content, the underlying models being used and the range of responses possible have been predetermined by the author. This is in contrast to the use of the computer applications described in the next strand, where the student has increasingly greater control over the knowledge being acquired and the method of acquiring it. While hard to prove, there appears to be an intrinsic appeal to the concept that as student control increases, the benefit of the learning experience with computers is enhanced.

Computer Applications

THE COMPUTER applications strand deals with a variety of applications of computer technology: word processing, manipulation of data and information in data bases, retrieval of information, electronic spreadsheets, and programming of languages. Perhaps the single most important criterion for the successful introduction to computer applications is the amount of time a student actually spends working with computer applications to solve problems. Students can be passive learners, essentially responding only to the presentation, style, and content in a particular application. However, students can learn more effectively by serving as active partners in an application, deciding which information to work with and what manipulations to perform on it and then instructing the computer how to do the manipulation. These applications also provide opportunities for the development of a wide range of critical-thinking and problem-solving skills.

As the sophistication of computers and software grows, it becomes increasingly difficult to classify certain software as an application program or as a programming language. In general, application programs are referred to as nonprocedural languages. While application programs have large vocabularies, they may be used to solve only a particular type of problem. A data-base manager, for example, is designed to manipulate large quantities of data, to change them, add to them, and to retrieve them. But a data-base manager is not an appropriate tool for managing the production of a large quantity of text, such as this document.

Procedural languages, in contrast, are very flexible in their use. A particular procedural language might be suitable for developing such diverse applications as a data-base manager, a word processor, or computer-assisted-learning materials. A vital skill for students to learn is the selection of an appropriate language, procedural or nonprocedural, for solving the problem on which they are working.

As shown in Figure 1, the use of the computer in an educational setting can be placed on a continuum from computer-assisted learning (CAL) to formal computer programming (with procedural languages), with applications programs (nonprocedural languages), such as word processors, electronic spreadsheets, data-base managers, and information retrieval systems, in the center. Underlying this continuum is the idea of student control over the computer. Within CAL, drill and practice represent an activity in which the student has little or no control over content or presentation strategies, and the student tends to be a passive receptor of information. Formal programming, on the other hand, puts the student more fully in control of the computer and its operation, because the student decides what the computer will do and what information it will use. Other types of application lie between these extremes of control. The level of control depends on the specific application.

Traditionally, most of the use of computers in schools has been concentrated at either end of the continuum, as there was little suitable software in the middle. As more application programs suitable for use in schools and simulation programs designed specifically to encourage student control become available, they will increase the opportunities for substantive use of computers throughout the curriculum.

Students should have experiences with computer applications in a wide variety of subject areas, all of which can be enriched through the appropriate use of the computer. Given a creative and cooperative situation among teachers, students in a programming class could develop and maintain a data base and reporting scheme for the athletics department or prepare educational games for lower grades. In addition to learning the specifics of an application or programming language, students taking part in these activities will have the opportunity for developing and applying a range of critical-thinking and problem-solving skills across curriculum areas.

In each of the general classes of application described in this section, there is a broad range of skills to be taught, and the software that can be used ranges from the relatively simple to the very complex. The particular subject areas to which an application is applied and the specific software that will be used will depend on the level of sophistication of the teacher and the level of development of the students.

▶ *Word Processing*

Word processors are achieving considerable success as an aid in teaching writing because they provide the opportunity to focus on the creative aspects of writing, such as putting ideas on paper and then editing them without rewriting each draft by hand. Students can therefore experiment more freely with organizing and editing their writing. This opportunity is an important factor in producing quality writing.

As hardware becomes cheaper and more prevalent in schools, word processing should become a readily available tool to be used for assignments in all subject areas.

The specific word processing skills to be learned include:

1. Touch typing skills
2. Text entry
3. Text editing (insert, delete, block move, block copy, block delete, search, and replace)
4. Use of spelling checkers
5. File handling (save, copy, delete, merge, append)
6. Print formatting (headers, footers, underline, font styles,[1] subscripts,[1] superscripts[1])
7. Preparation of form letters with variables[1]
8. Integration of word processors and data-base managers[1]

[1] For the more motivated or proficient student.

▶ *Electronic Spreadsheets*

Electronic spreadsheets, which were originally developed to assist with financial forecasting, can be used in a number of subject areas in addition to business education. For example, in a class project, heights, weights, and ages can be entered on a spreadsheet each quarter and the data manipulated to test hypotheses about the relationships among these variables. In social studies a spreadsheet can be used to record and manipulate census data gathered by the class, either from a field research effort or from government sources. Financial planning for a bake sale can also be done on a spreadsheet. With these figures students can test hypotheses about the relationships among expenses, prices, and profits.

The specific spreadsheet skills to be learned include:

1. Use of established spreadsheets to answer What if? questions and to test hypotheses
2. Design of spreadsheet formats
3. Use of formatting commands (copy, delete, insert, global versus local formats, row and column descriptors, table definition)[1]
4. Use of data entry commands
5. Construction of formulas
   a. Mathematical functions
   b. Logical functions[1]
   c. Financial functions (present, future value, and so on)[1]
6. Report formatting and printing
7. Spreadsheet file handling (delete, save, copy, merge)

In addition to these specific skills, use of spreadsheets also provides opportunities for learning general communication, problem-solving, and occupational skills, such as:

1. Reading and understanding documentation
2. Creating documentation

[1] For the more motivated or proficient student.

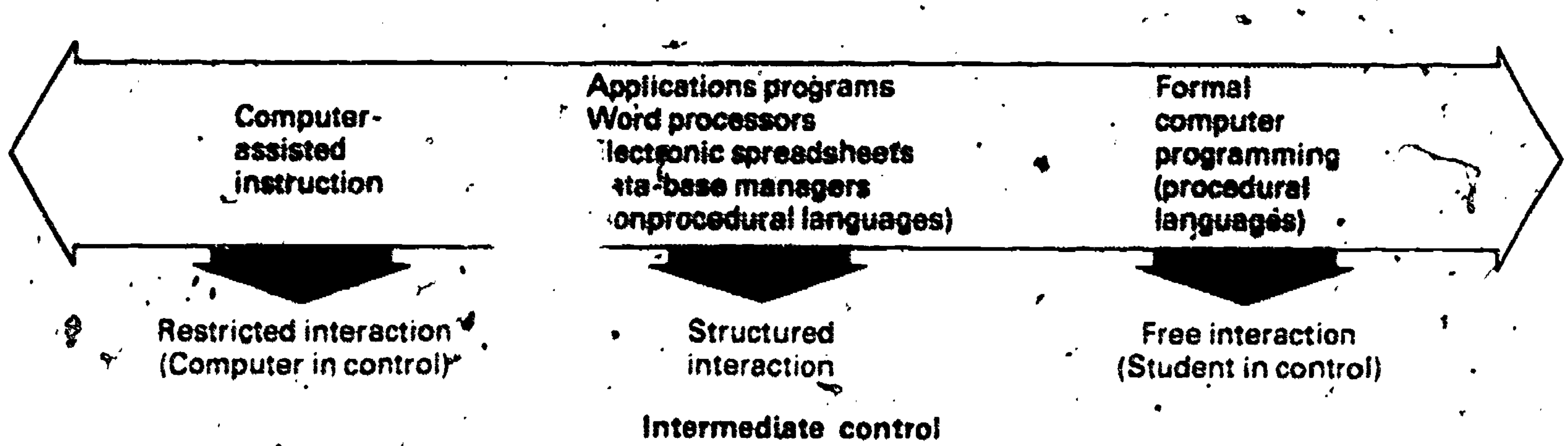

Fig. 1. Computer use and student control

3. Adhering to useful and appropriate conventions
4. Collecting, organizing, analyzing, and interpreting data
5. Testing hypotheses
6. Estimating and approximating
7. Recognizing that results are only as good as the initial data and the model used to analyze them

▶ *Data-Base Managers*

The ability to retrieve data in usable form is an increasingly important skill. The ability to collect, organize, analyze, and interpret data has widespread application across curriculum areas. It is also an important problem-solving skill in its own right.

For example, students could gather insects, research the data on them, and then enter the information into a data file. The information recorded for a specimen might include its family and order, color, place usually found, time of day found, and type of food preferred. Students could then develop simple hypotheses about insects and their environment and could use the data file to generate empirical information to test those hypotheses.

Students could also be responsible for setting up and maintaining data bases, such as names, addresses, and telephone numbers of the class or the class library and its checkout system. Maintenance of data and statistics for athletic events is an application that is of use to the whole school community and could be done as part of a regular class or as an extracurricular activity.

The specific data-base management skills to be learned include:

1. File design
   a. Fields: alphabetic, numeric, alphanumeric, fixed length, and variable length
   b. Records—fixed and variable length
   c. Sequential and random access files[1]
2. Use and design of data entry formats
3. File handling
   a. Save, delete, copy, revise, sort, merge, append
   b. Use of multiple files[1]
4. Report design and specification
5. Production of form letters[1]
6. Access to public and subscriber data bases

In addition to these specific skills, the use of data-base managers also provides opportunities for learning general communication, problem-solving, and occupational skills such as:

1. Using language precisely
2. Reading and understanding documentation
3. Creating documentation
4. Adhering to useful conventions
5. Collecting, organizing, analyzing, and interpreting data
6. Checking reliability and currency of data
7. Controlling, owning, and protecting privacy of data
8. Testing hypotheses
9. Estimating and approximating

▶ *Programming*

The application offering the highest level of user control over the computer is programming it to perform some worthwhile task. Programming puts the student in charge of the computer, providing a sense of power and accomplishment and enhancing self-esteem. In addition, programming may be an effective way to teach critical-thinking, problem-solving, and communication skills. These are important reasons for all students to acquire, at least at an informal level, some fundamental concepts of programming. However, not all students need to be taught programming in the formal sense.

Students should be provided with the opportunity to combine their knowledge of programming with that of some other content area. In mathematics and science the opportunities to do so are rather obvious, but they exist in other areas too. For example, students could write instructional materials for younger students. Such an exercise requires a thorough understanding of the subject matter to be programmed, some understanding of teaching strategies, and the ability to program in an appropriate language.

Students can also program a game such as ticktacktoe and are thereby free to use their creative abilities to make games of interest for their peers. Learning to program a robot can be an excellent introduction to both programming and geometry. In the fine arts, students can program both sound and graphics, again allowing for individual creativity. Students should also be given opportunities to write practical and useful programs for their own use and the use of their peers. For example, a program can be written to select at random one name from a list of volunteers entered through the keyboard or a program to maintain the inventory in the science laboratory.

When given as group assignments, such projects not only develop good programming skills but also foster cooperation, interpersonal relationships, and precise communication. In addition to providing these benefits to students, these projects foster cooperation among teachers in different content areas. In designing activities for teaching programming, it is impor-

[1] For the more motivated or proficient student.

tant that examples from mathematics and science be used sparingly. In a school with a full-fledged computer component, programming examples should come from all subject matters, and the design of appropriate activities should be a cooperative venture between disciplines. Thus, just as ethical issues of computing should be dealt with in social studies, so can simple problems from social studies be solved in a programming class.

Students could be introduced to programming first informally and later, as appropriate, in a formal fashion. The informal approach focuses on the students' using a computer to solve a variety of graphics and word problems in the context of problem solving or learning to "think about thinking." At an early stage students should be encouraged to experiment with suitable languages and a variety of problem-solving strategies.

For example, in an assignment to use a suitable programming language to create a picture containing a house, a tree, some flowers, a lawn, and the sky, students can adopt a number of strategies for completing it. Some students will carefully plan everything before working at the computer; others will go directly to the computer and begin creating parts of the picture, moving them about, changing size and shape until a pleasing effect is created. A majority might plan a small part of the assignment, write a program to accomplish it, plan a little more and program it, and so on until the assignment is completed. The "doers" are responding to the assignment in a concrete way, while the "planners" are approaching the problem in a more abstract manner. The important learning activity here is thinking about how to get the task accomplished and experimenting with various strategies because the picture itself could be more readily drawn without a computer or with a computer and a graphics tablet.

The formal approach to computer programming builds on the informal and deals with critical-thinking and problem-solving skills in a more rigorous and deliberate way. The focus is on learning these skills and their relationships to each other as abstract ideas. They are illustrated with more complex problems, perhaps using languages that are new to the student.

In addition to producing the picture described in the preceding paragraph, the task at the formal programming level might be to write an outline that shows the structure of a program in sufficient detail so that the program could be written by the author or by several different people. Formal programming skills may not be necessary for every student; and when teachers examine the educational uses of computers, they should consider the total curriculum and the wide range of student abilities, interests, and career goals.

The set of programming skills that may be taught are primarily language-independent, although some are not possible at all in some languages and others are much easier in some languages than in others. The specific programming skills to be learned are the abilities to:

1. Accept and use input from the keyboard, data files,[1] and peripheral devices, such as light pens and joysticks.[1]
2. Define and produce output in text, graphics, sound, and robot actions.
3. Define and use simple procedures and subroutines.
4. Use control structures for simple iterations (loops and recursions) and choices (branches).
5. Define and use numeric and string variables.
6. Use arithmetic operators.
7. Use Boolean operators and expressions.[1]
8. Design and use sequential and random data files.[1]
9. Design and use numeric and string arrays.
10. Design appropriate error trapping routines.

The order in which these skills should be introduced will clearly depend on the sophistication of the teacher, the experience level of the students, and the languages available in the classroom.

[1]For the more motivated or proficient student.

Problem Solving with Computers

COMPUTERS ARE not only tools for problem solving but also excellent tools for acquiring critical-thinking and communication skills. They offer the learner concrete experience with realistic situations in which problems can be formulated and solved with the use of a variety of thinking and communication skills. For this increasingly critical dimension of the curriculum, the software employed will usually range from simulations to application languages to procedural languages.

This strand describes various thinking and communication skills which can be used to solve problems and which can be taught with the use of computer applications and programming languages.

▶ Thinking Skills

Each of the following critical-thinking and problem-solving skills is important in its own right in many situations throughout the curriculum. Both programming and computer applications provide excellent vehicles for teaching and practicing these skills.

Describing the Problem

A critical first step in solving any problem is being sure that it is understood clearly enough to know when an acceptable solution is reached. Being able to describe it to the satisfaction of the person posing the problem is one way of checking that the problem is understood. In real situations the problems do not arise already formulated. More typically, one must analyze and evaluate the situation to discover the problem. Students can develop the skills needed to discover and formulate problems by working through assignments which define situations rather than through prepackaged problems.

Breaking Down the Problem

Once the problem has been clearly understood, the next step is to break it down into manageable pieces. Called top-down design, this powerful problem-solving procedure is applicable in many areas of human endeavor. The principle involves taking the problem, which might appear unmanageable, and breaking it down into a series of simpler subtasks. Each subtask can then, if still too complex, be broken down into still smaller steps. This process can be continued until the problem remaining at each step is either simple enough to be analyzed and solved or has a known solution.

For example, a research project might be broken down into four major stages: data gathering; data verification and organization; data analysis; and report writing. Data gathering might then be further broken down into designing the data collection form or instrument, selecting the sample to be surveyed or the information sources to be searched, and collecting the information. In a programming assignment each subtask then becomes a subroutine or procedure in the final solution.

Students may also be introduced to this breaking-into-manageable-parts technique for a wide variety of problems that are not computer-related. Noncomputer-related activities can be used to introduce the concept before it is used with any particular computer application.

Students may be asked to plan their "back to school night" classroom activities by taking the overall event and breaking it down into small, manageable pieces. Although the event in its entirety may seem difficult and unmanageable, everyone can accomplish some essential task that makes the whole event a success when it is broken down into small pieces (make the lemonade, put up art work, arrange the desks, and so on).

An outline of the activity is an excellent way to show the top-down nature of the solution. Drawing diagrams to show how the parts of the solution are related to one another and to the whole can also be a useful technique. Whatever method is used to record the solution, great emphasis should be placed on making it simple and easy to follow.

Developing Algorithms

An algorithm is a list of instructions or a recipe for performing some task. Most complex tasks involve developing an algorithm, or series of algorithms, that will lead to their successful completion.

Everyday life is full of algorithms: the directions for getting from home to school, the recipe for making cheesecake, and the instructions for disassembling a carburetor. Two important lessons arise from focusing on the use of algorithms. First, when developing an algorithm that is to be used to accomplish a new task, the developer must be specific, and the algorithm must not contain ambiguities.

Secondly, there is frequently more than one way to accomplish a task. People can become more efficient and effective in the conduct of their lives by building a library, or tool kit, of algorithms that can be used to avoid reinventing solutions to recurring problems. Since algorithms are so prevalent, many opportunities for developing and using them exist. Examples are preparing recipes; giving directions on how to get from one place to another; and writing instructions for checking out books from the library.

Anticipating Outcomes and Finding Unforeseen Results

As the various subparts of the overall solution are designed, it is important to test each subpart to see not only what happens when all goes according to plan, but also what will happen when the unusual occurs. Thus, the plan for a school's open house should address questions such as, "What if three people come with every student in the class?" if the planning is based on two, and, "What shall we do if it rains?" if some events are scheduled to take place outside.

In informal programming with a graphics language or with a robot, students should be encouraged to get up and walk around to simulate the action of the computer or the robot at all phases of program design. Part of the testing of their ideas should involve seeing what happens if, for example, unexpectedly large or small values for variables are given to a subroutine or procedure. Will it fail, or will it merely produce an unexpected consequence? Students may also design a program that produces a short instructional sequence. For each question asked, the students should anticipate at least two versions of the correct response and at least one incorrect response so that differential feedback may be given to the learner.

Generalizing a Problem Solution

An important ability related to breaking a problem into manageable parts is that of being able to generalize from the solution of one problem to the solution of another seemingly different problem. The ultimate success of the breaking-down strategy depends on the potential for arriving at subparts for which solutions exist or solutions can be easily generated. Students should be encouraged to build libraries or tool kits of various solutions that can be used in the future.

Students learning programming should describe in a notebook each subroutine or procedure they develop. The description should include why each was developed, exactly what it does, and what some future uses might be. Similarly, if students have used a data-base manager to support a particular learning activity, students should record how data were gathered and what features of the data-base manager were used. A systematic and deliberate teaching of how to avoid "reinventing the wheel" should stand students in good stead in many subject matter areas.

Testing and Debugging

Once the procedures or subparts of a problem have been designed, it is necessary to test them to be sure that they do exactly what they were designed to do. In programming, because procedures and subroutines rarely work the first time, the process known as "debugging" must be used. Models in an electronic spreadsheet must also be tested to be sure they produce the correct results. To debug successfully, one must first develop a hypothesis about what went wrong. Once a hypothesis is developed, it must be tested. This involves deciding what test to use, what input to give the procedure or financial model, and what output to expect from it.

Setting up hypotheses and tests involves the important skills of estimation and approximation. For example, if a programming routine that is to add together two numbers gives a result of five when given the numbers five and ten as input, it is safe to assume there is a bug in the procedure. It may be one of at least two errors: (1) subtraction is performed rather than addition; or (2) only the ones-digit portion of the result is printed as output. Hypothesizing that it is the first problem, the student makes up some examples and the output is calculated, assuming the procedure was subtracting instead of adding. Then the examples are run through the procedure to see if the predicted output is produced. If it is, then the error in the procedure is almost certainly discovered.

▶ Communication Skills

The ability to communicate clearly is a key skill for all students. Working with computers and application programs provides opportunities for learning to communicate with the computer and with other people.

The very nature of computers makes communication skills of paramount importance. Computers require very precise use of language because they cannot discern meaning from half-formed sentences, intonation, body language, and so on. Once the creative part of using an application program—designing the solution strategy—is completed, another important skill is required. The solution strategy has to be described in very precise terms, using the codes and conventions of the particular application or programming language and computer.

Errors in communication with a computer fall into two broad categories, lexical and syntactic. Lexical errors involve using words that the computer does not have in its vocabulary. Syntactic errors involve words that the computer recognizes but that are out of context or in an order that make no sense to the computer. These two are common errors in verbal communication also. Through interaction with different programming languages, students will come to recognize that to communicate well in any subject area, it is necessary to adhere to the vocabulary and conventions of that subject.

Communications with other people are important in all computer applications. In programming, to be sure that the problem to be solved by the program is clearly understood, the programmer and the person desiring the program must communicate clearly. When the program has been completed, it is important to write documentation that explains in very clear terms how it works and how to use it. Students should also be encouraged to work together on most applications, to share ideas and techniques, and to build the vital skill of cooperation.

Computer Science

THIS STRAND, computer science, describes content that is most likely not suitable for most students and should be offered as an elective. In general, it would be taught in a formal computer science course. It is important to point out, however, that much of this material overlaps that in other strands. The material from the computer science strand that is taught to any specific group of students should be dependent on their skills and interests, not on some kind of arbitrary division of content.

The specific computer science skills and knowledge to be learned by the more motivated or proficient student include:

1. Evaluate the use of, and differences between, procedural and nonprocedural languages, assembly and machine languages, compilers and interpreters.
2. Understand number systems used in computers.
3. Evaluate and use various data representation schemes.
4. Summarize data storage and access techniques for memory and disks.
5. Understand the basic architecture of computers.
6. Understand elementary electronic circuit design principles.
7. Apply Boolean algebra appropriately to the solution of problems.
8. Design, write, and debug an application program in a procedural language.
9. Apply specialized techniques such as sorts, searches, error trapping and recovery, and so on to the solution of a programming problem.
10. Understand the job market opportunities in computer-centered occupations related to programming.

Societal Impact of Computers

THIS STRAND, the societal impact of computers, should also be covered in a wide variety of settings and subject matter. While the proposed content of this strand is computer-related, it cuts across subject matter; the major topics of study have been chosen to reflect this fact. Many of these topics will have greater impact and relevance when dealt with in conjunction with some specific hands-on experiences with computers than they will have if dealt with in isolation. For example, while the ethical considerations surrounding software copyright should clearly be dealt with directly in the context of computer use, they also have a place in social studies units devoted to societal ethics in general and in business education.

The use of computers influences many areas, especially the following:

1. **Ethics and Values**
   a. Evaluate the implications of copyright laws.
   b. Evaluate ways in which access to computer systems and data bases can be protected.
   c. Interpret the issues surrounding the ownership, control, and accuracy of information.
   d. Summarize the arguments for providing public access to informaiton and for protecting individuals from invasion of privacy.
   e. Predict ways in which computers may eliminate jobs and change job skills and evaluate the effects of these changes on society.
   f. Describe ways in which computers can be used to commit crimes, and evaluate security measures designed to protect computers and the information they contain.

2. **Impact of Computer Technology on Lives and Careers**
   a. Predict ways in which computers and related technology will be used in the home.
   b. Predict ways in which computers and related technology can be used in schools.
   c. Predict ways in which computers and related technology will affect personal transactions such as shopping, banking, telephoning, and so on.
   d. Predict ways in which computers will be used in manufacturing, farming, merchandising, banking, and so on.
   e. Predict ways in which professionals such as managers, lawyers, scientists, accountants, writers, engineers, editors, and artists will use computers in their work.
   f. Evaluate ways in which governmental agencies, including police departments, health and welfare agencies, tax assessors' offices, the Internal Revenue Service, and so forth, use computers.
   g. Summarize aspects of modern life that would not be possible without the use of computers.
   h. Evaluate the impact of computers on the workplace.

3. **History of Computers and Computing**
   a. Describe the origins of number systems that are related to computing.
   b. Describe the development of calculating machines.
   c. Summarize the principal differences between generations of computers.
   d. Identify individuals who made key contributions to the development of computers.
   e. Evaluate the trends in computer development, and predict what computers will be like and used for.

4. **Technological Consumer Skills**
   a. Develop and use evaluat on tools for computer software.
   b. Develop and use evaluation tools for computer hardware.
   c. Evaluate the usefulness of specific hardware in the home.

d. Identify appropriate and inappropriate uses of computers.

5. **Vocational/Career Information**

a. Describe occupations that use computers as tools.

b. Describe occupations specifically connected to computers and computing such as programmer, software engineer, repair technician, systems analyst, computer operator, and data entry technician.

c. Evaluate the educational qualifications required for various computer-related occupations.

d. Evaluate and predict the impact of computers on the workplace in terms of organization of work and job skill requirements.

20

Publications Available from the Department of Education

This publication is one of approximately 500 that are available from the California State Department of Education. Some of the more recent publications or those most widely used are the following:

Administration of the School District Budget (1983)	$3.00
American Indian Education Handbook (1982)	3.50
Apprenticeship and the Blue Collar System: Putting Women on the Right Track (1982)	10.00
Arts for the Gifted and Talented, Grades 1-6 (1981)	2.75
Arts for the Handicapped Trainer's Manual (1982)	6.50
Bilingual-Crosscultural Teacher Aides: A Resource Guide (1984)	3.50
California Private School Directory	9.00
California Public School Directory	12.50
Career Vocational Assessment of Secondary Students with Exceptional Needs (1983)	4.00
Child Development Program Guidelines (1983)	3.75
College Core Curriculum: University and College Opportunities Program Guide (1983)	2.25
Computer Literacy of California's Sixth and Twelfth Grade Students (1984)	1.50
Curriculum Design for Parenthood Education (1982)	4.00
Guide for Vision Screening in California Public Schools (1984)	2.50
Handbook for Planning an Effective Mathematics Program (1982)	2.00
Handbook for Planning an Effective Reading Program (1983)	1.50
Handbook for Planning an Effective Writing Program (1983)	2.50
Handbook for Teaching Portuguese-Speaking Students (1983)	4.50
History-Social Science Framework for California Public Schools (1981)	2.25
Improving the Attractiveness of the K-12 Teaching Profession in California (1983)	3.25
Improving the Human Environment of Schools: Facilitation (1984)	5.50
Improving the Human Environment of Schools: Problems and Strategies (1979)	2.50
Improving Writing in California Schools: Problems and Solutions (1983)	2.00
Individual Learning Programs for Limited-English-Proficient Students (1984)	3.50
Instructional Materials Approved for Legal Compliance (1984)	6.00
Literature and Story Writing: A Guide for Teaching Gifted and Talented Children (1981)	2.75
Making Mealtime a Happy Time for Preschoolers (1983)	7.50/10
Manual of First-Aid Practices for School Bus Drivers (1983)	1.75
Martin Luther King, Jr., 1929-1968 (1983)	3.25
Mathematics Framework and Addendum for California Public Schools (1984)	2.00
Nutrition Education—Choose Well, Be Well: A Curriculum Guide for Junior High School (1984)	8.00
Nutrition Education—Choose Well, Be Well: A Curriculum Guide for High School (1984)	8.00
Nutrition Education—Choose Well, Be Well: A Curriculum Guide for Preschool and Kindergarten (1982)	8.00
Nutrition Education—Choose Well, Be Well: A Curriculum Guide for the Primary Grades (1982)	8.00
Nutrition Education—Choose Well, Be Well: A Curriculum Guide for the Upper Elementary Grades (1982)	8.00
Nutrition Education—Choose Well, Be Well: A Resource Manual for Parent and Community Involvement in Nutrition Education Programs (1984)	4.50
Nutrition Education—Choose Well, Be Well: A Resource Manual for Preschool, Kindergarten, and Elementary Teachers (1982)	2.25
Nutrition Education—Choose Well, Be Well: A Resource Manual for Secondary Teachers (1982)	2.25
Nutrition Education Today: Curriculum Design for Nutritional Knowledge and Food Use in California's Public Secondary Schools (1981)	2.50
Physical Performance Test for California, 1982 Edition (1984)	1.50
Planning Vocational Home Economics Programs for Secondary Schools (1983)	2.75
Preparing Food for Preschoolers (1983)	7.50/10
Preschool Program Guidelines (1983)	2.70
Raising Expectations: Model Graduation Requirements (1983)	2.75
Reading Framework for California Public Schools (1980)	1.75
Resources in Health Career Programs for Teachers of Disadvantaged Students (1983)	6.00
School Attendance Improvement: A Blueprint for Action (1983)	2.75
Science Education for the 1980s (1982)	2.00
Science Framework for California Public Schools (1978)	1.65
Science Framework Addendum (1984)	3.00
Statement on Competencies in English and Mathematics Expected of Entering Freshmen (1982)	2.50
Studies on Immersion Education: A Collection for U.S. Educators (1984)	5.00
Techniques for Preventing the Spread of Infectious Diseases (1983)	1.50
Time and Learning in California Schools (1984)	1.50
Toward More Human Schools: Exemplary Efforts in Self-Concept, Human Values, Parenting, and School Climate (1981)	1.75

Trash Monster Environmental Education Kit (for grade six)	23.00
University and College Opportunities Handbook (1984)	3.25
Visual and Performing Arts Framework for California Public Schools (1982)	3.25
Wet 'n' Safe: Water and Boating Safety, Grades 4–6 (1983)	2.50
Wizard of Waste Environmental Education Kit (for grade three)	20.00

Orders should be directed to:

California State Department of Education
P.O. Box 271
Sacramento, CA 95802-0271

Remittance or purchase order must accompany order. Purchase orders without checks are accepted only from government agencies in California. Sales tax should be added to all orders from California purchasers.

A complete list of publications available from the Department, including apprenticeship instructional materials, may be obtained by writing to the address listed above.

A list of approximately 100 diskettes and accompanying manuals, available to member districts of the California Computing Consortium, may also be obtained by writing to the same address.

83-154 (03-0442) 78783-300 12-84 10M

84 78783