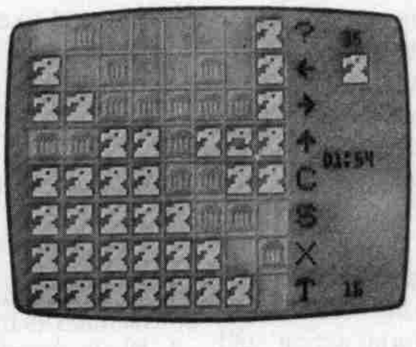
By David V. Loertscher

ANALYZING MICROCOMPUTER

THERE IS an alternative to purchasing commercially produced microcomputer software: school media specialists and teachers can create their own. There are skeptics who say there will never be any high-quality, locally produced, educational microcomputer materials because it takes too much time and effort to produce them. However, these critics should remember the *Foxfire* phenomenon, the creative filmmaking by high-school students, and the success of programmers who have become millionaires in the arcade computer business, most of them under 20 years of age.

For three years, I've taught microcomputer courses to teachers and library media specialists, and have observed many excellent products they have developed. I am enthusiastic about library media specialists producing microcomputer software locally. Already, imaginative and creative library media specialists are joining with capable student programmers to produce programs to fulfill local curriculum or management systems needs. Recent news items provide evidence that librarians are using programs they have devised to handle overdue materials, compile bibliographies, store inventory records, organize computer-generated catalogs, and more. Another microcomputer program produced locally is one which interfaces with commercial media such as videotapes, filmstrips, and slides.

David V. Loertscher is Program Coordinator, Instructional Resource Education, University of Arkansas, Fayetteville



This article is directed to school library media specialists and teachers who want to start producing their own microcomputer software for students' use. One of the best ways to begin is first to understand the rudiments of programming and then to compile a notebook, which I call a "bag of tricks," of ideas for microcomputer programs. Any number of features can be incorporated into a program at any stage of production.

To start such a notebook, I suggest taking an idea from one program and another from a second and then using these accumulated ideas to produce a program that fills your needs. This practice of copying the good features of commercial programs and eliminating questionable ones is often used by advanced programmers and is well known to library media specialists as the old cut-and-paste technique. Professional programmers, for example, have many pre-written subroutines at their disposal; if a list of names or words must be alphabetized, they can use pre-written sorting routines. There is no need for them to write regularly used routines over and over again.

I want to emphasize here that I am not recommending that librarians try to crack the codes of commercial producers and steal parts of their programs. The sole purpose of the notebook is to catalog good, usable elements for homemade software programs.

Tricks of the Trade

My bag-of-tricks technique consists of one looseleaf notebook and several microcomputer disks, which contain useful subroutines. The notebook should be divided into two sections: the first section is for your collection or listing of desirable features for incorporation into microcomputer programs you plan to create, as well as a list of pitfalls to avoid. The second section will contain pages, each with one desired program feature entered at the top and brief notes on how to program it listed below. For example, if your program presents a page of text, and you want to command the student to press the space bar to get to the next page, your notebook page of commands, comments, and sample program will resemble the one below, adapted from *Modular Computer Lesson Design* by Paul M. Roper and David V. Loertscher (Hi Willow Research & Publishing, 1982). These basic program commands for the task will guide users through the lesson: (see box opposite)

SOFTWARE

Procedures for Notes

Every time you preview or use a microcomputer program, watch for programming features you do and don't like. Make brief notes about these features in section one of your notebook, and in section two, make notes on how to program these desirable features—if you can figure out how they were done. Often you can add a programming idea to the notebook by perusing journals such as *Popular Computing, C.A.L.L., Apple, Creative Computing*, etc. To get more ideas or to refine your own, you might also attend a local microcomputer users group to ask questions. Books in this area are beginning to appear on the market. Locally, there may be inservice training sessions conducted to acquaint teachers with computer literacy. One goal of these inservice sessions is to help teachers create a product to use immediately with their students. The commercial skeleton programs, into which teachers may insert their choice of spelling words or curricular terms, plus a few ideas from their own "bags of tricks," should get them started in a positive way. Many times, a teacher may program a very simple lesson in BASIC and use it to great advantage in a teaching sequence even if it is not of commercial quality.

Ownership of the material is often enough motivation to do a better-than-average job of presenting a desired concept to students. This is not to excuse poor work, but it can motivate teachers to use this relatively new technology with a minimum of frustration. If inservice workshops don't provide a useful product, teachers are likely to forget much of what they have learned in a matter of days.

Desired Task: To get the user to advance to the next section or page of the lesson.

Commands for APPLE OR PET (using line numbers for each program code line)

```
10 PRINT "PRESS SPACE
   BAR TO CONTINUE"
20 GET A$
30 IF A$ = CHR$(32)THEN 50
40 GOTO 20
50 REM REST OF PROGRAM
   HERE
```

Commands for TRS-80

```
10 PRINT "PRESS SPACE
   BAR TO CONTINUE"
20 A$ = INKEY$
30 IF A$ = CHR$(32) THEN 50
40 GOTO 20
50 REM REST OF PROGRAM
   HERE
```

Comment: CHR$(32) is the ASCII code for the space bar. Any specific character on the keyboard may be used by using the ASCII code for that character (an ASCII code list can be obtained from a microcomputer reference book).

Sample Program for CLS:

```
10 HOME
20 PRINT "WHAT IS 2+2?"
30 PRINT: PRINT: PRINT:
   PRINT: PRINT
40 PRINT "PRESS SPACE
   BAR TO SEE AN-
   SWER"
50 GET Z$
60 IF Z$ = CHR$(32) THEN 80
70 GOTO 50
80 PRINT : PRINT : PRINT
90 PRINT "4 IS THE AN-
   SWER"
```

Software—Critical Reviews

CLOCK. rev. ed. (Apple II, 48K, disk) Hartley Courseware, Inc. 1982. $39.95. Teacher's manual.

In today's world of digital time pieces, is it still necessary to teach children to read clock hands? If you answer yes, then *Clock* is an excellent choice of a program to accomplish this. This program has many exemplary programming elements and the content is adaptable to local situations. It is obvious that a good educator and a good programmer created this software.

There are four objectives of the program. Students can manipulate clock hands to set a prescribed time in two different ways: they can type in the exact time from a preset clock face, or they may just watch the clock face and the given correct time as a tutorial or review exercise.

Directions to the student are easy to read and understand. The positive reinforcement provided has enough variety to keep it fresh, and the negative reinforcement will not discourage users, but encourage them to try again. The management system for monitoring students is excellent, and provides for entry of 40 student names. In addition, a list of errors is compared to the correct responses. This provides the teacher with instant feedback on what kinds of errors the students made so that supplementary instruction may be given right away. The teacher's manual is well written and succinct. The high-resolution, black-and-white graphics are simple but effective.

This program is a good example of the type of microcomputer software that can make a positive contribution to education. Its combined elements of teacher control, student control, ease of use, and utilization of computer characteristics make it exemplary.

The Game Show. (Apple II, 48K, disk). Computer-Advanced Ideas, Inc., 1981. $39.95. Teacher's manual.

If you enjoy the "Password" game by Parker Brothers or the television show of the same name as an instructional/motivational game, you will find this skeleton program attractive. The game can be created for two teams. Each team has a partner on the computer screen who is a regular participant on *The Game Show*. The regulars, named Joe and May, help their teammates guess a mystery word by providing helpful hints or clues.

While six sample games are supplied on the diskette, the program is really designed to enable teachers to create games for instructional purposes. Spelling words, vocabulary words, names, or recall terms can be used. Clues can be one word or a short phrase, synonyms or antonyms. Students in the third grade or higher can easily create more games to challenge other students. Instructions in the program and in the manual are clearly stated. Individual games can be added, changed, listed out on the screen or dumped to paper (for an answer key), saved on disk, or deleted. The best feature of *The Game Show* is the ease of creating new games and game disks because, unfortunately, after about six plays, the students are likely to tire of the format. It should be used sparingly. Despite the limitations stated, the program is well worth the purchase price.

Arcademic Skill Builders. (Apple II+, 48K, disk). Developmental Learning Materials. 1981. $39; $220 set of 4. Teacher's manual. Flash cards.

The purpose of these six skill builders (two of six games were reviewed) is to combine the excitement of an arcade game with drill in addition, subtraction, multiplication, and division. The player must defend home base by shooting and destroying some invader (alien space ships, large green slime blobs, meteors, tanks, or alligators). The invaders approach home base at a constant pace, threatening to destroy the base unless annihilated. Each invader contains a math problem which must be solved by the student, whose answer is typed into the home base figure, then the space bar is pressed to shoot. If the answer is correct, the alien is eliminated, but a shot with an incorrect answer is a miss, and the alien continues to advance. The excitement is intense, the time pressure is great, and the player's mental and manual dexterity is taxed.

Positive features are many. The teacher can exert some control over problems. Addition, subtraction, multiplication, and division problems may include 0-3, 0-6, or 0-9 factors. Total playing time can be set from one to five minutes. Speed of the invaders' progress can be slowed or increased. The number of hits and misses is recorded on the screen and charts are provided for a progress analysis by the teacher. To further enhance the game, detailed suggestions are given, flash cards provided, drill sheets included, and a specific plan to teach strategy is recommended. The central idea of the game is instantaneous computing of the math fact—no time for finger counting under this pressure! Two of the games use mixed problems for added difficulty.

Without a doubt, these drills are some of the very best on the market. The graphics are eye-catching. The control over game time, game speed, and data recording is excellent. Even key manipulation can be controlled for right- or left-handed players. Paddles

Sample notebook pages

Skeleton Programs—Positive features

1. Minimal programming knowledge needed
2. Easy editing when revisions are necessary
3. Student uses a product disc only, not the authoring program, which the teacher keeps. This provides protection against student changes or the mutilation of the master
4. Good system of menus in the final product so that users can get in and out of the program quickly
5. Good instructions so that little individual help is needed when students are working alone.
6. Users can escape the game at any point.
7. The sound effects can be turned off or on.

Skeleton Programs—Pitfalls to avoid

1. Assumes too much programming knowledge for user to create lessons.
2. Master disc must be used for operation of the program.
3. Poor motivation; limited possibilities for new programs.
4. Modifications to the skeleton are so tightly controlled that local adaptation is very limited.

Drill Programs—Positive features

1. Attempts to build motivation into the drill.
2. Drill, as a memorization device, could have key words disappear at random.
3. Uses animation in math problems to show math operations.
4. When the screen must be blank, the program informs the user of the reason why.
5. Provides an analysis of student errors.
6. Uses appropriate vocabulary for intended audience.
7. Supplies a variety in positive and negative reinforcement.

Drill Programs—Pitfalls to Avoid

1. Graphics detract from the focus of the drill.
2. Games without educational value are used as reward.
3. Vocabulary is too difficult for target audience.
4. Method of entry for correct answers is unclear.
5. Lengthy answers are required from students with limited typing skills.

Graphic Tricks—Positive features

1. Dancing words are used as clues.
2. Screen wipes (top down, bottom up, middle out, diagonal, etc.)
3. Essential details are provided for critical interpretations.

Graphic Tricks—Pitfalls to Avoid

1. Slow graphics. Test the program time on intended audience.
2. Graphics overshadow the instructional message.
3. Illegible high resolution character fonts. High resolution character fonts should be legible for every letter or redesign.

may be used if available. The teacher's guide is very useful for planning, charting progress, and following up with math activities.

There are some problems. Some educators might be uncomfortable with the use of an arcade-game format integrated with an instructional skill. The problems in the games are limited to single-digit numbers, and no provision is made to increase the probability that a missed problem will appear more frequently as the game progresses. The time pressure may also be disturbing to some students. But, if used properly, these games (for students from grade one up) can serve as good motivational devices for memorizing math facts.

Genesis. (Commodore/PET) Greenwood Software. 1981. cassette. $30.

Do you want to create a flashcard drill without any pictures? This skeleton program is designed for the teacher who wants to create drills to test recall of specific information or arithmetic facts connected to some lesson or unit being taught in the classroom. The questions must be limited to a maximum of 71 letters. Variations on answers can be entered by the teacher so that synonyms, full or abbreviated names, or variant spellings are judged correct by the computer. The computer, as programmed, may administer the drill either sequentially or by pulling questions at random. Questions missed by the student remain in the pool and will be pulled again until answered correctly.

No programming skill is required to create drills except very basic computer literacy. The menu system used is outstanding: corrections, additions, deletions, and other editing features are readily available. The program is unprotected and may be copied within a school so that every teacher can have access to this drill.

Since this program is an excellent tool for teaching teachers how to create their own homemade drills, it will be particularly useful in an inservice computer literacy course.

George Earl's Reading in Literature. (Apple II, 32K, disk). Kvitle Kourseware. 1978. $29.95.

Can students use the microcomputer to familiarize themselves with famous passages from literature? That is the goal of this carefully programmed tutorial/drill. Eleven different literary passages are presented, including *The 23rd Psalm, Annabel Lee, The Bill of Rights, Casey at the Bat*, and *The Gettysburg Address*. The object is to guess the last word of each phrase in the passage which begins, "Four score and seven years ???" If the response is an "a," the student is rewarded with 50 "perfect"s dancing on the screen. If the student can't remember the word, four guesses are given at any letter in the word. As in Hangman, the user is punished for every wrong guess with a colorful cubist graphic that shrinks as each wrong answer is given. If the graphic disappears completely, the correct word dances on the screen and another chance at guessing the phrase is provided, but now the complete passage is presented twice. The first time, the full phrase to be tested on the next screen is given. That is the clue. The second time around, the student is expected to enter all the words in each phrase without help. A final score indicates how well the student performed.

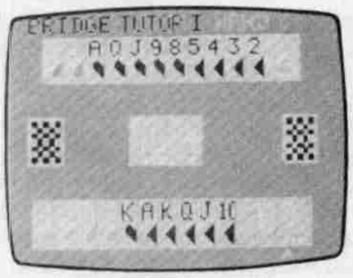
The programming tricks used on this disk could fill a notebook. There are fascinating screen wipes, boxes around normal text, dancing words, and interesting title graphics. But some major questions arise. Why test recall of only the last word of a literary phrase? Why present a colorful graphic on the screen when it has no relation to the words at the bottom? This program almost succeeds as a memory aid, but these are serious flaws, and it is not recommended for purchase.

States, States2 (from Elementary Disc. Vol. 3, version 4.6) (Apple, 32K, disk). Minnesota Educational Computing Consortium. 1981. $39. Teacher's manual.

Since every intermediate student in the U.S. makes some attempt to learn the states' names, to identify each by shape, and name their capitals, it makes sense to create a microcomputer program that would help them. Two programs reviewed, *States* and *States2*, have this objective, but a number of problems limit their usefulness. The *States* program is a very simple word drill; it has no graphics. Students may choose a section of the country on which to be drilled or may be tested on all the states. The format is: "What is the capital of Nebraska?" The response must be spelled correctly, but occasionally the program will allow a one-letter misspelling and indicates "you were close." Most students could program this type of drill by knowing a few simple computer commands, although computer analysis of student input would be a little more difficult.

States2 shows the outlines of states in high-resolution graphics. States are to be identified and the names typed in, correctly spelled. Unfortunately, the graphic is so small on the screen that an accurate state outline map is not presented, particularly for the Northeastern states. *States2* allows no deviation from the correct spelling, unlike the *States* program.

Library media specialists can devise ways to improve these two programs: allow a variety of student responses such as A,B,C, or abbreviations as well as full spellings, or the use of dots rather than lines to produce the shapes of the states.



Guinness World Records Problem Areas in Math. (TRS-80 or Apple, 32K, 4 disks). Society for Visual Education, Inc., 1982. $225. Teacher's manual. Skill sheets.

Using children's fascination with world records as the motivating force, this package combines both tutorial and drill in addition and subtraction to correct specific skill deficiencies (adding large numbers, adding numbers in jagged columns, subtraction problems with regrouping, and subtraction problems with zero in the minuend). An elaborate management system for the teacher provides control over the various types of problems and problem difficulties. In addition, detailed reports of the progress of up to 30 students in various class groupings is provided.

The two main segments of this program are a mastery section and an application section. In the mastery section, either numeral or story problems are presented, and if the student misses the answer, a tutorial is provided. If three out of five problems are done correctly, a chase through a maze game is the reward. The application section requires the solution of three out of five problems in order to play an elaborate "free the Guinness world record holder from the terrible space grogs" game.

The major flaw in this set is the inordinate amount of time required to do a lesson or play a game. For example, it takes 38 seconds to do a single number problem, 66 seconds to do one story problem, 8 seconds to get a reward for a correct response, 6½ minutes to set up just three math problems. The reviewer figured that five problems plus the game could be done in 12 minutes, working at maximum speed. The manual says that during testing, students took an average of 15-25 minutes to do one session (five problems and a game).

On the application disks, response time is so slow on the part of the computer that the grogs hold a decided advantage. The time lapses stem from a decision on the part of the writers to create a wonderful graphics show. This they have accomplished very well; the program is extremely colorful. But providing a 15- to 25-minute graphic show in exchange for only three to five correctly solved problems is a questionable use of students' time.

Another programming-related problem was the number of times the disks malfunctioned. Misreads, total failures, and bizarre tutorial runs occurred often enough to question its programming structure rather than to blame a bad disk.

Turning to substantive problems, the regrouping procedure in subtraction is not how students are taught to perform on paper. When borrowing is necessary, no superscript number appears. Also, when the user must borrow, the original number which has been borrowed from disappears from the screen rather than being crossed out with a digit placed above it. There is a nice feature, however, using animation when numbers are repositioned on the screen from a tutorial on the right to the main problem on the left.

The management system of the program was well done. Thorough records, student groupings, provision for students to exit and return later, and teacher control over lessons are excellent. The additional skill sheets which accompany the disks are the best feature of all. Interesting problems and puzzles containing world record clues provide excellent followup to the games and the tutorial.

Before purchasing this package, a thorough on-site test is recommended to see if the time problem and malfunctions mentioned are manageable and if the tutorial style fits in with the method teachers are currently using. A reviewer will also want to assess the reading level on the story problems to see if slower readers can learn from the program.

Introduction to Computers. (series). cassette (automatic and manual). color. 4 strips with 4 cassettes. range: 60-68 fr., 11-12 min.. Teacher's guide. Society for Visual Education, Inc. 1981. #A517-SATC. $145 series; with 2 microcomputer disks for TRS-80 or Apple (32K), $205.

Using an urban high-school class setting, computers are introduced in four segments. The first filmstrip, "The Purpose of Computers," shows four teenage students discussing some common fears and concerns about computers, such as whether they are likely to control our lives. The second filmstrip presents an explanation of hardware and software, introduces various terms such as input device, memory, processing unit, output device, and explains how these components process information.

"The Impact of Computers," third in the series, provides numerous examples of the effect computers have on society, such as control and switching of telephone calls, price-tallying in grocery stores, computer crime, and the possibilities for careers in the computer industry. The series ends with a discussion of how a computer is programmed. The series also explores program analysis, flowcharting, coding, running a program, and debugging it. The two computer disks which accompany the filmstrip set are designed for optional followup. The disks can be used by individuals or by a single teacher with an entire class.

Despite the fact that the narrator contradicts himself about whether or not computers make errors, and that occasionally the visual does not match the narration, the set may be used for a very basic introduction to computer literacy. No grade level is indicated on the materials.

Unfortunately, the programs have a great many technical and substantive flaws; just a few of these flaws will be itemized here to illustrate what happens when programs are not carefully planned, tested, or debugged. On disk one, the title graphic takes up much too much time (the copyright notice alone stays on the screen for 26 seconds before moving into the program). The direction to move from one screen to another says "press the space bar to continue"; any key that is pressed will advance the program. When the reviewer was given a choice of answers A-G, a "4" was entered which was interpreted as a "C" answer. Later, during a simulation, an incorrect response sent the program back to a previous menu without an explanation. The simulation was supposed to demonstrate how an airport runway is cleared, but the action appearing on the screen was never explained. Although the filmstrips are acceptable as introductory material, the package is not recommended because of the many problems encountered.

Questions & Answers

One often hears that most of the educational software created for microcomputers is either mediocre or worthless. Such an overgeneralization, if widely accepted as truth, will hinder the use of this technology in our nation's schools before it can succeed in helping to educate our children. Many say that they will not buy software until quality materials are plentiful, but if we all wait until this is possible, the producers of microcomputer software may abandon the educational market as an unworthy investment. Educators should be assured that there is enough quality software available at present to warrant the purchase of microcomputers in every school. It is necessary to sort through all the new products and select carefully. The materials are available. Librarians and school media specialists must continue to review and demand higher quality materials of producers, just as they do for audiovisual materials and books.

In the meantime, we can learn to create our own software. Now let's suppose that you begin your notebook, start the programming section, and suddenly realize that you are not cut out to be a programmer. You ask yourself, "What good has it been to start this notebook?" Several answers to that question come to mind. First, the more you examine commercial software, the better critic and judge of quality you will become. Second, many school media specialists will have opportunities to communicate with a computer programmer. If you have acquired some computer literacy and can communicate clearly the type of programming features you like and don't like, the job for the programmer becomes much easier and hours of exasperating revision of programs can be saved. In addition, like most creative activities, there is a great deal of satisfaction in producing your own quality educational software. Try it! ◀

SUGGESTED READING

Bruce, Phillip & Sam M. Pederson. *The Software Development Project: Planning and Management.* John Wiley, 1982.
Designing Instructional Computing Materials (manual and 2 diskettes) 2 vols. MECC, 1982.
Writing Support Materials for Instructional Computer Programs. MECC, 1982.

DISTRIBUTORS

Computer Advanced Ideas, Inc., 1442A Walnut St. Suite 341, Berkeley, Calif. 94709.
Developmental Learning Materials, One DLM Park, P.O.Box 4000, Allen, Tex. 75002.
Greenwood Software, 1214 Washington, The Dalles, Oreg. 97058.
Hartley Courseware Inc., Box 431, Dimondale, Mich. 48821.
Kvitle Kourseware, 15510 Heimer Road, San Antonio, Tex. 78232.
Minnesota Educational Computing Consortium (MECC), 2520 Broadway Dr., St. Paul, Minn. 55113.
Society for Visual Education, Inc., 1345 Diversey Parkway, Chicago, Ill. 60614.

Illustrations: TRS-80 color computer programs: Roman Checkers; Mega-Bug; Bridge Tutor