Experiences from a life sciences laboratory: Using a paired-associates task as a teaching heuristic for programming instruction

J. W. WHITLOW, JR.

Rutgers University, Camden College, Camden, New Jersey

A method is presented for using a paired-associates learning task as a model for teaching programming skills to psychology students with limited computer experience.

Enthusiasm for using microcomputers to teach undergraduate psychology majors how to conduct psychological research is widespread, as evidenced by the papers in this issue. There are, however, divergent views as to how best to bring student and computer together, with one source of divergence being the issue of whether or not to teach students how to write computer programs. Some researchers (e.g., Hewett, 1987) explicitly reject the usefulness of programming instruction. I am sympathetic to many of the concerns that Hewett expresses, especially the fear that students may become so immersed in programming esoterica that any psychological content will be lost. At the same time, I think that the most memorable way for students to understand both the capabilities and the limitations of the computer is by doing some programming of their own. The question, then, is whether it is possible to teach students how to write computer programs for research in a reasonable amount of time and without becoming bogged down in programming per se.

This paper describes one approach to teaching psychology majors how to program microcomputers in BASIC. The essential feature of the approach is to use a paired-associates experiment as a model or template of a research project. Our initial experience with this approach suggests an affirmative answer to the question raised above. Students can be taught how to program the computer to conduct the paired-associates task in a reasonable amount of time and are able to generalize their programming skills to a variety of other kinds of tasks without further instruction.

BACKGROUND

The psychology department at the Camden College campus of Rutgers University is relatively small, graduating about 35 majors per year. The overwhelming majority of our majors are female, and about one third are women returning to school to complete degree work

that was interrupted by marriage and raising children. Most of the students plan careers in education, health care, or counseling services. Their quantitative skills, including computer literacy, are weak, and they lack confidence in quantitative reasoning abilities. Surveys taken over the past 2 years indicate that less than 25% own computers, nearly half (45%) claim to have never used a computer, and most report some computerphobia. Our students are, nonetheless, intrigued by computer usage and would like to learn more about computers, as reflected in the unanimous view that computers are important for psychology.

In 1985, our department received a grant from the New Jersey Department of Higher Education (DHE) to establish a microcomputer-based laboratory in psychology and integrate computer usage into the curriculum. The grant allowed us to purchase nine IBM PCs (one of which was a PC/XT) and two Tandy 1000 PC compatibles. The relatively large number of computers allowed us to teach computer usage in classes with no more than 2 students sharing a computer. In our experience, it has been important to limit the number of students working at a single terminal, because many students simply watch unless they are forced to interact with the machine. A passive approach severely limits how much active problem-solving practice the students get and how much they learn about actually using the computer.

We offer a one-semester course on computer applications in psychology. This course is designed to cover three main areas: (1) programming computers to conduct experiments, (2) using "canned" statistical analysis programs, and (3) using computer simulations as analytical and theoretical models of behavior. For programming instruction, which takes up half of the course, we needed a means of rapidly teaching something about computer programming to students who did not have strong computer literacy and who had limited opportunities to practice on the machines.

THE PAIRED-ASSOCIATES MODEL

Using a paired-associates model as a template for experimental programs is an idea taken from Deni's (1986) book on writing programs for psychological research.

Send reprint requests to J. W. Whitlow, Jr., Psychology Department, Camden College of Arts and Sciences, Rutgers University, Camden, NJ 08102.

Deni used a paired-associates task as his first example of program construction, and he developed a number of elaborations of it over the course of the book. Deni, however, introduced a variety of programs in addition to the paired-associates task. We have used the paired-associates task for all of our exposition because it provides a very general template for many types of research tasks. (The general utility of the paired-associates task as a model of psychological research is interesting. There was a time when paired-associates learning was *the* model of human learning. Perhaps the durability of paired-associates as a model of learning was due to a confusion with its role as a model of research practice.)

One of the first things that is illustrated by the paired-associates task is that the typical sequence for the flow of information in a computer program (input, processing, and output) does not map easily onto the requirements for using the computer as an experimental control device. In experimental usage, the focus must be shifted from the computer program to the subject, for whom the experimenter provides input and who produces output that is then processed. For example, in the paired-associates task, the subject is presented with a list of pairs to study (input) and then a series of test trials in which the subject tries to give the response term to a stimulus-term probe (output). The set of test responses are then analyzed for correctness and reported in an appropriate summary (processing). Thus the logical sequence for experimental applications is input, output, and processing. It turns out that this sequence also provides a useful guide for sequencing the introduction of elements of programming. In the following sections I will elaborate on this point for each of the three components.

Input

The input phase of a paired-associates task consists of presenting to the subject a list of word pairs for study. The programming for list presentation is divided into two parts, emphasizing a simple list display in the first part and introducing the use of arrays and pointers in the second part. The simple list display requires only six programming concepts: (1) clearing the screen, (2) displaying the text, (3) controlling cursor location, (4) using loops, (5) assigning data to variables using the BASIC READ command, and (6) distinguishing integer from character variables. Of these, the one that is most difficult for students to grasp is that of the loop, which in BASIC uses the syntax of a FOR...NEXT pair. As illustrated in Program A in Table 1, loops are used in two ways at this stage. They are used for the repetitive sequence of READING and PRINTING the word pairs and to control timing through a software loop. By the end of this first phase of programming, students can present a list of word pairs in a designated position on the screen and can control the presentation rate.

The goal of the second part of the instruction on list presentation is to develop a more general method for

Table 1
Three BASIC programs for Presenting a List of Stimulus Response Pairs for Study, Showing the Evolution of Code from Simple Use of a READ Statement to Use of a Pointer to an Array

A. CLS	Clear the display screen.
FOR I=1 TO N	Start of display loop.
READ A\$,B\$	Get stimulus and response terms from DATA statements.
LOCATE 10, 30	Position cursor.
PRINT A\$,B\$	Display stimulus and response terms together.
FOR J=1 TO LIM	Start of timing loop.
NEXT J	End of timing loop.
CLS	Clear screen at end of word.
NEXT I	End of display loop.
B. CLS	Clear the display screen.
FOR I=1 TO N	Start of display loop.
A\$=STIM\$(I)	Get stimulus and response terms from arrays.
LOCATE 10, 30	Position cursor.
PRINT A\$,B\$	Display stimulus and response terms together.
FOR J=1 TO LIM	Start of timing loop.
NEXT J	End of timing loop.
CLS	Clear screen at end of word.
NEXT I	End of display loop.
C. CLS	Clear the display screen.
FOR I=1 TO N	Start of display loop.
II=PNTR(I)	Get element location from pointer.
A\$=STIM\$(II)	Get stimulus and
B\$=RSP\$(II)	response terms from arrays.
LOCATE 10, 30	Position cursor.
PRINT A\$,B\$	Display stimulus and response terms together.
FOR J=1 TO LIM	Start of timing loop.
NEXT J	End of timing loop.
CLS	Clear screen at end of word.
NEXT I	End of display loop.

presenting stimulus displays by using pointers to select items from an array of possible stimuli. The use of pointers also allows us to introduce questions about the control of presentation order and to develop the programming for "shuffling" a stimulus series. This section requires introducing only two new programming concepts, the notions of an array and of a pseudo-random number generator. Students first change their programs so that the stimulus and response lists are READ into arrays, rather than into simple variables; then they present the list by using an index number to step through the array, as shown in Program B of Table 1. The next step is to set up a pointer array and present the list by using an index number to pick up event identification numbers that are used to locate events for presentation from the stimulus and response arrays. Program C of Table 1 illustrates this step. The final step is to use the pointer to present list items in varied orders. This step includes a descrip-

tion of the BASIC RND function, and students are given a shuffling routine for randomly rearranging a pointer array.

Output

After learning to program the study phase of the paired-associates task, students are taught to program the test phase. The test phase involves presenting the stimulus terms and collecting the subject's responses, which are then scored. This component introduces the use of keyboard input, conditionals, and subroutines.

The first step, requesting a response from the subject, is accomplished using the INPUT command. The subject's response is compared to the correct response using the IF...THEN syntax of BASIC, and the outcome of this comparison is used to provide feedback. Scoring and the presentation of feedback provide a good place for introducing the concept of "error trapping." As soon as a subject is allowed to respond, students encounter the difficulty of designing "forgiving" programs that can catch errors, such as typing in uppercase or using numbers instead of letters. Subroutines are introduced through the problem of presenting stimulus terms in the test phase in an order different from that used in the study phase. This is most easily done by making the shuffling routine a subroutine that is called prior to list presentation and again prior to the test presentations.

Processing

We concern ourselves with two aspects of processing the data collected from a subject. First, students write program code for calculating means and standard deviations. This exercise serves as a way of stressing the importance of initializing variables and as an important illustration of recursive computation, which involves computational methods that often differ from the student's normal approach to a problem. No attempt is made to teach students statistical programming, since they learn to use statistical packages later in the course.

The second aspect of data processing is that of storing data in a disk file. This section introduces the elementary concepts of disk I/O for sequential files. Students learn to create a file name, open a file, send data to the file, close the file, and read data from the file. The programming code for accomplishing these actions is relatively simple in Microsoft BASIC.

EVALUATION

By way of evaluating the paired-associates model as a teaching heuristic, I can point both to what the students were able to accomplish and to what they either failed to accomplish or did not accomplish well.

The most significant accomplishment was that students who had only the programming training described in the preceding sections were able to write programs for other kinds of research. Each student in the course was assigned

a separate project, to maximize the degree to which students would be individually responsible for writing a program. All students managed to complete their projects. The projects included (1) the generation and storage in a file of all possible CVCs, for use as stimulus materials; (2) a computerized presentation of the Bem Sex Role Inventory (Bem, 1974), with scoring of responses and a subject classification; (3) a computerized presentation of the Stunkard Restraint Scale (Stunkard & Messick, 1985), with scores calculated on cognitive restraint, disinhibition, and hunger sensitivity; (4) a sentence memory program; (5) a reading readiness program that asked readers to fill in the blank for a simple sentence by selecting from among three alternatives; and (6) a language learning program that presented foreign-language equivalents to English words. All of these programs required some amount of generalization from the paired-associates model, although the amount of overlap varied from very little, as in the language learning program, to a substantial amount, as in the computerized questionnaires. However, from our point of view, using the paired-associates model was very successful. Students with little or no programming skills were able to produce simple programs that will be of actual use to them in the settings they are likely to encounter outside the classroom. In addition, as they became more confident of their skills, many of the students embellished their programs to incorporate special visual or auditory effects or to present aesthetically pleasing displays.

Despite the significant accomplishment that emerged from using the paired-associates model, I do not want to leave you with the impression that our students had become proficient in programming. Proficiency was not the goal; we simply wanted to instill a degree of confidence and develop some solid elementary programming skills. There were many specific aspects of programming that students did not master. Three areas appeared to be particularly difficult: (1) the use of arrays to hold data, (2) the use of pointers, and (3) the use of subroutines. To some extent, these difficulties may simply require more attention by the instructor during presentation in class. The difficulty with using arrays to hold data may, however, represent a proactive interference effect from the first to the second stage of programming instruction for list presentation. In the first stage of programming, lists were presented directly out of DATA statements, using READ and a single variable (see Table 1, Program A). Perhaps as a consequence, some students treated the placing of items in an array as equivalent to the listing of items in DATA statements, whereas the two operations are fundamentally different.

SUMMARY

The paired-associates learning task has been used as a model with which to teach experimental programming skills to students with limited computer experience. The

model provides a general framework both for introducing programming concepts and for designing programs for a variety of other applications. In our experience, students are able to transfer their knowledge reasonably well from the paired-associates task to these other applications. Thus we feel that this approach is one way to include programming instruction in a relatively efficient manner as part of teaching undergraduates how to use computers in research.

REFERENCES

- BEM, S. (1974). The measurement of psychological androgyny. *Journal of Consulting & Clinical Psychology*, **42**, 155-162.
- DENI, R. (1986). *Programming microcomputers for psychology experiments*. Belmont, CA: Wadsworth.
- HEWETT, T. (1987). The computer as a tool in instructional computing: Students as software architects. *Behavior Research Methods, Instruments, & Computers*, **19**, 123-127.
- STUNKARD, A. J., & MESSICK, S. (1985). The three-factor eating questionnaire to measure dietary restraint, disinhibition, and hunger. *Journal of Psychosomatic Medicine*, **29**, 71-83.