Programmed Learning

The State-of-the-Art for Training Practitioners

Ralph A. Drekmann

Today's training specialist is facing a difficult manpower problem: he has to hire the skills he can get and develop them into what his organization needs. To help him bridge that gap, he will rely on training.

One of the most promising training methods is programmed learning. Here is a discussion that may sound familiar to you:

> "Let's pay more attention to the individual learner and give him a chance to experience achievement."
>
> "Okay, but let him learn by doing the thing. He may think he knows it, but we can't be sure until he tries."

Programmers discussing training strategy? No, that's quoting Quintilian and Sophocles in free translation.

Then programmed learning isn't really something new? That's right. You can trace its beginning all the way from Plato's Socratic dialogs to Pavlov's study of conditioned reflexes in dogs, from Thorndike's stimulus-response studies to Pressey's multiple-choice testing machine and Skinner's ping-pong playing pigeons.

So why all the excitement about programmed instruction and teaching machines? Because the re-discovery of programmed learning principles may by one of the most significant contributions to human learning in this century. For the first time, behavioral psychologists have taken the results of their scientific experiments out of the laboratory to apply them directly to human learning.

What are some of the psychological principles underlying programmed learning? Most importantly:

Operant conditioning
Reinforcement

Ralph A. Drekmann

Manager, Learning Systems Design, Communications Division, Equitable Life Assurance Society of the United States. Has training methods development responsibilities. B.B.A., City College of New York; M.B.A., New York University; doctoral studies in behavioral sciences and business, City University of New York.

Discrimination
Behavior shaping

Animal Analogy

These principles have to do with habit formation. Here is an over-simplified example: Suppose you want to teach your dog a trick. You want him to retrieve a ball out of the water. By chance you see him run to pick up a stick. He is responding to something, he is showing operant behavior. That is, his behavior operates on the environment.

Right away you give him a piece of Yummy. It's a stimulus. He likes it, feels rewarded and tries again. You have given him immediate reinforcement. Note that your dog must be active. You cannot reinforce his behavior unless he is busy doing something.

Next you introduce the ball and give him that reinforcing Yummy stimulus only when he picks up the ball. Now you are using selective reinforcement to make him react to the difference between ball and stick. You are trying to bring about conditioned discrimination.

By designing a logical, systematic learning sequence, you can eventually train your dog step-by-step to retrieve the ball out of the water. You are shaping his behavior. You are building a new skill. If you help him maintain this behavior, we can observe it and will conclude that he has learned.

You can see that animal trainers have practiced certain principles of programmed learning for centuries. These principles can also be applied to training the human animal. In fact, Brethower and Rummler of the University of Michigan have pointed out that in your daily work with people, you are probably using operant conditioning right now, unwittingly, as a management technique. If you have shaped employees' behavior by deliberately encouraging only desirable practices, you have used operant conditioning.

How do programmers apply these principles to programmed learning? They have designed instructional sequences, called programs, with special characteristics. These training materials are carefully planned to combine typical features:

1. Step-by-Step Build-up
2. Active Response
3. Feedback
4. Self-Pacing

Let's look at each of these features briefly.

1. Step-by-Step Build-up

A well-constructed program is organized into clearly defined portions called frames. A frame may range in size from several pages of an overview chapter to the smallest step in a multiplication exercise.

Frames are sequenced in what the programmer perceives to be a logical order. If tryouts of his program show that trainees will learn a process more efficiently in an "illogical" order, he will use that tested order as the more effective teaching sequence.

Frames are built one upon the other in steps of increasing complexity. An earlier notion that all material to be learned must be broken down into small steps did not prove very useful. Instead, the programmer must find the optimum step-size by testing his frames, as they are written, among his intended audience.

2. Active Response

After studying the one specific point presented to him in a frame, the learner must respond. He may have to analyze a diagram, discriminate between typical examples, or form a concept. In this way he is actively engaged in the learning process. He learns by doing.

An earlier "rule" of programming suggested that all active responding should be observable behavior. For example, answering out loud or completing statements in writing. But humans seem to do a great deal of covert responding, hidden but very active. Many learners appear to carry on a silent but incessant dialog with themselves. They let thoughts race through their minds, manipulate concepts, attempt verbalizations, and keep answering questions to themselves. These too, are active responses and must be recognized and acknowledged by the programmer.

3. Feedback

A well-written program will guide the learner through problem-solving exercises which will reveal to him step-by-step that he is answering the questions correctly and finally that he has solved a problem correctly. In addition, the learner will find the correct answer to each frame printed in the program. This will give him a chance to check how he is doing. If he wants to, he can test himself and correct his own work. The printed answer will either confirm that he is right or show that he must correct his response.

At one time programmers thought that such feedback should always follow the frame right away. It would then serve as immediate reinforcement for the learner. But finding out that he made an error was not always a rewarding experience. So, the right answer in the program did not necessarily give reinforcement.

As a result, programmers became pre-occupied with the design of programs in which the learner would rarely make a mistake. They wanted programs to be as error-proof as possible. However, many adult learners would only once in a while check for correct answers. They didn't seem to care about immediate confirmation. They became so engrossed in their exercises and action projects that they rejected any interference. All they wanted to do was solve a problem and still another problem. Never mind some of their errors on the way. After completing the program they too demonstrated that they could perform the task just as well. They had achieved mastery; they had learned.

From these observations programmers concluded that immediate feedback is not always necessary or desirable. They have found out that confirmation is not synonymous with reinforcement. Many adult learners do think: "Let me make my own mistakes. And get off my back, so I can try this by myself."

Training specialists are usually looking for this positive problem-solving behavior in trainees. And they want them to be able to cope with their own mistakes. To be effective, programmed learning materials must therefore modify some of the features developed in the learning laboratory to adapt to practical needs and conditions.

4. Self-Pacing

Since the learner can check his answers without waiting for an instructor, a program can be both self-instructional and self-pacing. Of course, there is nothing new about a self-teaching workbook. It was already popular when Thomas Edison wrote his manual "Telegraphy Self Taught."

Many learners prefer to work their way through a course at their own speed. The fast learner can go through it in a breeze and the slow learner can take his own good time. Both should be able to perform the training objective after they have completed the program.

Trouble is that some trainees find it difficult to finish a program on their own. Programmed learning may be self-teaching but it is not self-administering. Some time control and selective supervisory monitoring is often helpful. Moreover, if study time varies too much, it may conflict with established training programs. To introduce programmed learning in an organization you must use commonsense and suggest reasonable completion schedules.

Program Development

These four features: Step-by-step build-up, active response, feedback, and self-pacing combine to form a repeating learning cycle. As each segment of knowledge is presented to the learner in frames of increasing complexity, he is required to respond to it in some way, and can then check for feedback to see how he is progressing. On his own, he can speed up this cycle or slow it down, as desired.

How is a program developed? Briefly, here are some of the essential steps used in the process of programming:

1. Examine existing system.
2. Investigate environmental factors.
3. Define trainee's job.
4. Establish need for program.
5. Analyze training task.
6. State training objectives.
7. Write final performance test.
8. Specify course pre-requisites.
9. Determine program design.
10. Select suitable media.
11. Write test frames.
12. Write teaching frames.
13. Conduct individual tryouts.
14. Revise program.
15. Validate program through field-test by group.
16. Revise, publish, install, and monitor program for next edition.

Questions to be Answered

Let's review some of these steps. The programmer should question:

1. Under what conditions is the program to be used. Is it a hostile environment where line management gives only lip service to training and thinks it's a nuisance? Does top management fully support the training effort?

2. Does the existing system reward the completer or is there a penalty for the trainee who does the job exactly the way it is taught? Does the training program direct trainees one way while their supervisors direct them in another?

3. What is the trainee now doing? What isn't he doing? What is his present repertory? Is it only a minor deficiency that keeps him from doing what you want him to do? Can he, in fact, do it, but doesn't want to? What are his needs?

4. If it is a case of "can do, but won't do," it's not a training problem. Maybe there is a need to improve communications, working conditions, compensation, or leadership. But there is no need for programmed learning materials.

5. Exactly what is the task the learner must accomplish? What are the typical steps in its performance? How does a master performer do it? What experiences do we observe in the field, on the firing line, at the gut level?

6. How do you define in measurable terms the specific behavior you can observe when the trainee demonstrates mastery? Does management accept these terminal behavior specifications as useful performance objectives?

7. Does the final test simulate real life conditions and the actual job as closely as possible? Is it a valid test?

Does it measure what it is supposed to measure?

8. What must the trainee be able to do before he is admitted to the course? What pre-test will serve best to screen out those who have not yet met the prerequisites for taking the program?

9. What teaching strategies will be most effective? Should we use a linear format, a straight-line tell-and-test routine? Do we want to accept the learner's constructed response, such as a written answer, as a receipt that our "goods" have been delivered? Or does this particular training problem call for a fast track in the program? Or for branching into extra drill or refresher loops? Should we use a multiple-choice format and accept selective responses? Can we develop enough plausible choices? Will some combination of these techniques give us a more effective program to let the learner reach his objective as quickly and efficiently as possible?

10. What medium shall we use to communicate? Should the learner look at videotapes, movies, film strips or slides? Listen to records, audiotapes, or lectures? Read a text? Are teaching machines suitable for this training job? Is computer-assisted instruction practicable for this particular task? Can a multi-media combination perhaps give us an even better simulation of the real life task? Can we employ other training devices, such as roleplay, to get the learner into the act of simulating the job for which he trains?

11. How can we create and develop a series of increasingly difficult task-simulation tests to measure the gradual change in behavior, the behavior shaping we want to engineer? How can we make these tests as close to real-life as possible so training will transfer to job?

12. Can we develop task-simulators and real-life examples to prepare a sequence of as few learning steps as necessary? Can we develop exercises which will guide the learner quickly to making critical discriminations and solving problems until he can pass each test frame?

13. How can we set up as realistically as possible tryouts with individual members of our intended trainee population to observe how they work their way through the program? Are the instructions in the program clear for proper use? Did we overteach?

14. Do we feel like revising the whole program because we tend to overlisten to the tryout learner? Does a re-sequencing of materials really improve the program? Did we provide enough practice? Can we trim verbiage to come up with a lean program?

15. Does the program teach what it is supposed to teach under actual field conditions? What are the problems of administering the program to groups in the field? How does the program fit into the on-going training effort? Is support equipment and material available when needed? What are the trainer's needs and attitudes?

16. How can we monitor the program's use, after it has been installed, through opinionaires, routine feedback from reports, or personal observation? What must we do to maintain the skills our trainees have acquired?

This long list of questions is only a sample of the programmer's analytical approach. He is a nosey fellow who wants to find out about things. He is not pleasant to have around, because he can ask a lot of unanswered and uncomfortable questions. Perhaps you would like to ask some of these questions about the task you are now working on.

This review of how a program is developed will now help us in spelling out a definition of programming:

Programming is a systematic process of developing and validating instructional sequences designed to change learner behavior in specific measurable ways.

Key Ideas

In programmed learning, five key ideas stand out:

1. Systematic analysis
2. Behavioral objectives
3. Task simulation
4. Testing and validation
5. Focus on the learner

First, programming means a systematic, organized approach to the analysis and effective coordination of all elements and influencing factors of a training problem.

Second, programmed learning is goal-oriented. A program should separate "nice to know" from "must know." One goes out, perhaps to be recommended as enrichment material; the other goes into the program, because it is necessary for goal achievement.

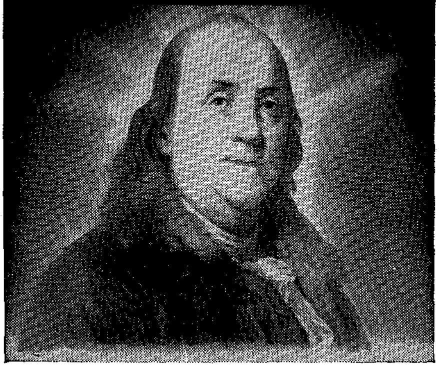
Third, before we send our astronauts into space, we want them to experience as many contingencies as possible in a task simulator. Before you send your trainee into the cold world to face your precious customers, you want him to handle many real life situations.

Fourth, testing and validating will give you vital feedback for accomplishing the programming task. It is at the core of the programming process.

Fifth, programming puts renewed emphasis on the learner. You are selling ideas and he is the consumer. He is the final authority on whether you have communicated. And to serve him is the programmer's ultimate goal.

Programming is an Art

Are all programs developed in this



systematic way? Far from it. We are only looking at guidelines. And they sound much simpler in theory than they are applied in practice. The problem is that humans are so much more than animals. Programming practitioners have convinced themselves that trainees are not learning like dogs or pigeons. For the behavioral psychologist this means going back to the laboratory for reappraisal. That does not mean his approach is not valid. It means that new questions have come up, but no definite answers. As in all science, there may only be an answer and an answer beyond that. Programmed learning has no special claim to solid scientific foundation, as it is practiced now. It is above all an art.

No Cure-All

What are the advantages of programmed learning? In hundreds of published studies there is evidence that programs will reduce training time, will reduce training cost, and will help increase the effectiveness of training. Any disadvantages? Yes. You still cannot make a trainee learn, if he does not want to learn. Some learners do not like to be led by the nose—ever. Some like to have a tutor present at all times to lean on when in doubt. Some miss the outside pressure to complete the program, if left to their own devices. Some training problems do not seem to lend themselves to a programming approach. Other techniques may work better. If so, it does not make sense to program for the sake of programming. Programming is no cure-all.

Who is using programmed learning? Today almost every major organization working with people and concerned about manpower development is making some use of programmed learning. Dr. Ofiesh, an early proponent of the programming method, has published 35 of the many case histories he collected from all branches of the Armed Forces, from a roster of leading industrial companies, banks, insurers, government agencies, giant retailers, and other organizations. Many of these reports tell of remarkable success through judicious application of programmed learning materials.

A Process, Not a Gimmick

The important thing to remember is that a program is *not* a product, is *not* a medium! A program is a process. Programming is not the re-writing of training manuals or textbooks to cover prescribed subject matter in a gimmicky way. Programming is a training method, a pedogogic strategy.

Alert training specialists who recognize that all training is basically communication have taken programming concepts and applied them to other communication jobs. For example, management consultants like Dr. Odiorne have used programming principles to propose "management by objectives."

Progressive educators at the Oakland Community College in Michigan have applied the same systematic development and evaluation in programming to define, design, produce, and implement all elements necessary for an entire institution of higher learning. All performance specifications were identified and defined in behavioral terms. They spell out the knowledge the student should demonstrate at the end of the course. Course content is sequenced meaningfully and based on a flexible time schedule. The student knows what is expected of him and he is kept informed of his progress. Criteria tests serve to evaluate his achievement of interim goals. An array of media is employed to present materials which allow active responding, give feedback, and permit reasonable self-pacing. The entire complex is learner-oriented, from time schedule to physical design of "learning laboratories" which make up the college plant.

Viewed and applied in this way, programmed learning reveals its true scope and great potential. Programmed learning techniques can be used to develop adjunct training materials to support your present training efforts. Programming can help you develop a year-long training course, re-design your total training program, or even your training center. Properly applied, programmed learning can make a valuable contribution towards more efficient and more effective training for your organization. It will help you solve some of your urgent manpower problems.

This article will appear in The Dartnell Personnel Handbook and the Dartnell Sales Managers Handbook.

Suggested Readings on Programmed Learning

1. Brethower, Dale M., et al. "Programmed Learning, A Practicum," Ann Arbor Publishers, 1965.

2. Glaser, Robert (ed.). "Teaching Machines and Programed Learning, II: Data and Directions," National Education Association, 1965.

3. Hilgard, E. R. "Issues Within Learning Theory and Programmed Learning," Psychology in the Schools 1: 129-39, 1964.

4. Jacobs, Paul I., et al. "A Guide to Evaluating Self-Instructional Programs," Holt, Rinehart and Winston, Inc., 1966.

5. Lumsdaine, A. A. and Glaser, Robert (eds.). "Teaching Machines and Programmed Learning. A Source Book," National Education Association, 1960.

6. Mager, Robert F. "Preparing Instructional Objectives," Fearon Publishers, 1962.

7. Mechner, Francis and Cook, Donald. "Behavioral Technology and Manpower Development," Basic Systems, Inc., 1963.

8. Murphy, John R. and Goldberg, Irving A. "Strategies for Using Programmed Instruction," *Harvard Business Review,* May-June, Vol. 42, No. 3, pp. 115-137, 1964.

9. Ofiesh, Gabriel D. "Programmed Instruction: A Guide for Management," American Management Assn., 1965.

10. Pipe, Peter. "Practical Programming," Holt, Rinehart and Winston, Inc., 1966.

11. Rummler, G., et al. "Managing the Instructional Programming Effort," Center for Programmed Learning for Business, University of Michigan, 1967.

Sources of Current Information

1. *Contemporary Psychology.* American Psychological Assn., 1333 16th Street, N.W., Washington 6, D. C.

2. *Educational Technology.* Educational News Service, P.O. Box 508, Saddle Brook, N. J. 07663.

3. *NSPI Journal.* National Society for Programmed Instruction, Trinity University, 715 Stadium Drive, San Antonio 12, Tex.

4. *Programed Learning Bulletin.* Center for Programmed Learning for Business, Bureau of Industrial Relations, University of Michigan, Ann Arbor, Mich. 48104.