Journal of Behavioral Education, Vol. 1, No. 2, 1991, pp. 235-251

Programmed Instruction: What It Is and How to Do It

Ernest A. Vargas[1,2] and Julie S. Vargas[1]

Accepted: January 7, 1991. Action Editor: R. Douglas Greer

Five behaviorological features of programmed instruction are outlined: behavioral objectives, reinforcement, activity rate: high and relevant, successive approximation, and mastery progression. Each of these topics is explained, and examples are given to illustrate the differences between programmed instruction and the more common 'transmission' model of teaching. A final section provides suggestions on specific techniques of successive approximation.

KEY WORDS: behaviorology; instructional design; programmed instruction; successive approximation; verbal behavior.

INTRODUCTION

Most people have heard of programmed instruction, but most people also think of it as a fad that died. It is true that as a term "programmed instruction" is no longer a popular buzzword in education, but its technology remains a cornerstone of good instructional design. Indeed, one might even claim that instructional programs succeed to the extent that they are programmed, whether or not their designers are aware of the principles from which the technology was originally derived. (A good early source for discussion and references on programmed instruction is Lumsdaine and Glaser, 1960. A later review, which pays special attention to the historical development of teaching machines, is Benjamin, 1988. Another history of

[1]Professor of Behaviorology, Division of Foundations, University of West Virginia, Morgantown, West Virginia.
[2]Correspondence should be directed to Ernest A. Vargas, Educational Psychology, University of West Virginia, Morgantown, West Virginia 26506-6112.

1053-0819/91/0600-0235$06.50/0

programmed instruction with special emphasis on Skinner's contribution is Vargas and Vargas, 1991.)

THE SCIENTIFIC FOUNDATIONS OF PROGRAMMED INSTRUCTION

Programmed instruction stems from the science of behaviorology. The science addresses the interactive relation between behavior and the milieu in which it occurs. Its explanatory framework, or paradigm, is selectionist (Vargas, E. A. 1991a). Selective factors operate upon behavioral variation to produce a shift in behavior that in turn changes the characteristics of the selecting milieu (Skinner, 1981).

Any activity of an organism affects its environment. The consequences that ensue increase or decrease the probability of subsequent actions occurring again. Stated in the most simple form then, events that follow actions have a selective function over the likelihood of actions with similar effects occurring in the future. In education, specific forms or topographies of behavior may be selected by the kinds of consequences made contingent upon them. The class of actions that produces similar consequences is called an *operant.*

Prior events that are paired with an operant, that is the action-consequence pair, gain control over responding. These antecedent events thus initiate an operant, and control the probability of its occurrence. Loosely speaking, we learn to respond in situations in which actions have in the past paid off. The addition of the antecedent event plus the two-term relation of the operant, results in a three-term contingency relation. Due to the vast number of pairings that occur, the same operants in a person's repertoire may relate to any of a number of antecedent events. For example, the operant "lifting a latch" may come under control both of latches of boxes and of gates. Other processes, such as induction or discrimination, extend or limit the control of antecedent stimuli or events.

Two-term and three-term contingency relations can operate without human intervention. But some kind of social group is needed for four-term contingency relations in which behavioral contact with events is *mediated* by the actions of others. If instead of lifting a latch myself, I ask you to do it, the consequence of an open gate is mediated by your behavior. The four-term contingency relation where behavioral control with events is mediated through behavior, is called *verbal behavior.* Two other characteristics define behavior as verbal: One, verbal behavior must have no necessary mechanical or geometric relation to the terminal consequation of the four-term behavioral relation. Unlike lifting a latch, *saying* "open the gate, please" does not physically operate the gate. Two, the mediating consequating action must occur because a "verbal community" established that action. People lift latches when asked to do so only because someone taught them to behave appropriately to that set of sounds. In addition to selection by the physical world, behavior is shaped and governed through verbal control, most of it through the process known as instruction. (For further description of the four-term contingency characteristics of verbal behavior see Vargas, E.A. 1991b; for a plausible reconstruction of how a verbal community develops see Skinner, 1986; for the theoretical explication of verbal behavior see Skinner, 1957.)

For the most part, how programmed instruction stems from behaviorology will be dealt with only indirectly, for this article examines the nuts-and-bolts of programmed instruction and not its relation to the science, but a brief discussion is necessary with respect to one of the critical components of programmed instruction–the frame. A frame is the basic analytic unit with which programmers work. Programs, for example, are talked of in terms of number of frames or sequence of frames. A thorough discussion of the frame is beyond the scope of this paper, but roughly speaking, it is an operant embedded in a four-term contingency relation that depends upon verbal behavior already-in-place for its successful execution. It is technically speaking a relational autoclitic (see Skinner 1957, pp. 333-340). The student is shaped to compose an answer, not simply to select one. Some frames may demand answers that are intraverbals, for example either sequelics or duplics, but good programmers typically avoid constructing responses that are merely paired with verbal stimuli, without further meaning to such a pairing, or that merely depend upon the formal properties of a prior verbal stimulus for their execution (see Vargas, E.A., 1986, for a definition and discussion of intraverbal behavior). The structural characteristics of a frame may thus vary, for example, its size in terms of verbal stimuli, or the amount of composition required. But all properly designed frames have in common the shaping of verbal activity in which the student is an interactive participant with the program.

INSTRUCTION AND PROGRAMMED INSTRUCTION

Education consists of constructing new repertoires, most of which are verbal. Educational curricula are built from records of the cumulative experience of prior generations; in large part what others have said or written about events, and in this way attempt to facilitate effective action when the student comes into contact with those events.

Instruction consists of the technologies through which education occurs. All instructional technologies address the control of student behavior. From motivation to attention, from enhancing self-concept to encouraging creativity, pedagogy consists of sets of actions by which desired outcomes, with respect to student performance, may occur. All instructors arrange instructional material in some sort of sequence that, presumably, facilitates learning. Most often, they sequence that material according to its inherent logic. But the resemblance between conventional classroom instruction and programmed instruction ends with sequencing material.

Before the programmed instruction movement of the 1960's, most approaches to pedagogy dealt primarily with presentation or "transmission" of "information", and they are still more common than behavioral approaches. The typical educator assumes that if the material is sequenced well and presented well, the instructional job has been done. If a student does *not* learn, the transmission model holds the student at fault. The student is failed. Or what is as bad, the current educational system passes the student along with less than full mastery, increasingly less prepared for each succeeding stage of the educational curriculum. Cumulative ignorance results.

Programmed instruction proposes just the opposite: If the student does not learn, then the instructional materials need to be revised. But more than reworking of the materials is involved. Programmed instruction addresses the controls over the teacher as well as those over the student. The instructional setting must be designed so that the teacher is also shaped to take the right actions. Programmed instruction consists of the teacher coming under control of the interaction of student behavior with specific features of the instructional setting. The cybernicity of programmed instruction distinguishes it from other instructional technologies: The student interacts with materials designed by an instructional designer, and the student's success, or failure, at each step of the instructional process shapes the teacher's revision of materials or design of future ones. The design effort is guided by the framework of the science combined with the effects of prior engineering efforts.

ENGINEERING PROGRAMMED INSTRUCTION

In this paper we outline a few principles and techniques for designing programmed instruction. But such exposition should only be construed as a beginning. For someone to be truly skilled, a knowledge of the basic science is necessary, especially its extension to the analysis of verbal behavior. One would not expect, on reading one article on the principles of suspension and stresses and strains, to be able to build quality bridges. For any major project, such as the George Washington or Golden Gate bridges, one would expect team work involving professionals with years of training and experience. The dynamics of behavioral events pose problems at least as complex as those in the physical and biological domains. To design an effective educational system requires more than taking sentences and cutting out little spaces that students can fill in. It requires the design of a special milieu for both the student and the teacher that continually changes as the student does. That requires teamwork between experts with training in the content of instruction and experts in the basic science (especially in verbal behavior) who have experience in designing instruction and watching students as they progress through programs.

Writing Programmed Instruction: General Principles

These principles apply to all cybernetic programs of instruction, and follow from the science. They are labeled: Behavioral Objectives, Reinforcement, Activity Rate: High and Relevant, Successive Approximation, and Mastery Progression. Specific techniques for executing one of the general design principles, Successive Approximation, then are described.

Behavioral Objectives

Programmed instruction starts with the behaviorological assumption that students bring a wide variation of behavior to any instructional setting. They must therefore be handled individually. Two general design goals, contradictory on the surface, must be achieved with this behavioral variation. First, student repertoires must *converge* to an identical repertoire–all shaped to a minimum mastery, similar in level and form. Second, at the same time, behavioral variation must also be shaped to *diverge* even further–individual differences must be recognized, strengthened, and even accentuated.

To set a minimum mastery focusing on the individual student, objectives are stated behaviorally–in terms of what the student should be able to do by the end of training. Instead of saying "This program teaches basic composition in the style of Bach chorales," an objective would be phrased "When you complete this program you should be able to write the remaining three voices in Bach chorale style given the soprano, alto, tenor, or bass line so that there is no voice leading that would have been considered an error in the Baroque era." Such objectives establish a common competency in all students moving on to later levels of a curriculum.

While adhering to a standard, programmed instruction promotes diversity. First, it encourages students who work faster than other students to go at their own pace. But rate of progress is not the only characteristic in which variety of behavior, or if speaking of a specific student, individuality of a repertoire, can be shaped. The richness of student backgrounds provides the capital by which a cybernetic program strengthens behavioral variety. Competence in basic skills, while requiring some similarity in student performance by the end of a program, provides the building blocks for creative differences. To use our music example, all students must adhere to Baroque voice leading. They must write each part so that, for example, no two voices go from one note to another in parallel fifths at any point in their melodies. But within that structure, the more facility one has with Baroque voice leading, the easier it is to come up with alternatives for unusual and interesting parts for soprano, alto, tenor, and bass. A firm foundation in basics thus promotes, rather than hinders diversity and creativity. Objectives for programmed instruction rarely include the word "creativity" because of its ambiguity, yet they enhance creativity by helping the teacher pinpoint the skills which make creative behavior possible. (As one reader of an earlier version of this article put it, "You can't play the blues until the chord progression evokes contingency-shaped behavior".) By writing behavioral objectives instead of vague goals or a list of topics that a course will "cover", the instructor must consider seriously what students need to be able to do to perform competently in a field and to do so in a unique manner. [Objectives can also include the aesthetic, emotional factor in instruction. See for example, Greer (1980).]

Reinforcement

A reinforcer increases the probability of the class of behaviors it follows. By definition, it always works. If a certain postcedent event does *not* strengthen behavior, that event is not a reinforcer for that behavior in that situation. Instructional designers use events that usually function as reinforcers to try to establish reinforcement for all of the problems a student must do to gain competence. If the consequences do not work, that is, students do not continue to work enthusiastically, the instructional designer must change the conditions that establish the consequences or change the consequences themselves until a good reinforcer is found. Most programmed instruction uses "getting the right answer" as "reinforcement". This presupposes both that the student will be right most of the time and that being right will be enough of a positive consequence to keep the student working. As with any "reinforcer", such an outcome is not inevitable, and the program designer should be prepared to design other kinds of reinforcers into an instructional system if being "right" is not sufficient. One computerized programmed instruction system (used primarily with inner city remedial students) adds points for completing lessons, typically at 85 percent accuracy. (This system is produced by New Century Education Corporation, Piscataway, New Jersey. Cost per student gain is discussed in Weinstock, 1984.) At various point levels, students receive certificates of accomplishment and a photograph of themselves holding their certificate. They also earn a free period. Although not every student cashes in his or her points, for some students the points are incentives. Getting answers right is not enough incentive to keep them working carefully one period each day in the center. More valuable points were used with delinquents in a prison setting. Students earned points by completing lessons, and spent them on everything from renting a private room to accessing the program library (Cohen & Filipczak, 1971). These two cases are unusual. Most programs use "feedback" alone, as reinforcement. Instructional designers need to be aware, however, that feedback (or "confirmation of results" as it is sometimes called) may not be reinforcement for some students.

Unlike the traditional classroom, which gives rewards only loosely contingent upon behavior, programmed instruction provides tight contingencies. A consequence is provided immediately following each response. The more immediate the delivery of a reinforcer following a prior action, the greater its effect. Timing is critical. Just a fraction of a second may occur before a student begins to think of something else, fiddle with a pencil or pen or keyboard, look away from the lesson page or screen, and so on. A reinforcer delivered at this point loses its punch and may even strengthen non-educational behaviors, though through "backward chaining" some effect ensues on the desired action. The immediacy of reinforcement prevents other behavior intruding between the appropriate action and the delivery of reinforcement. At the very least, to qualify as immediate, a consequence for an action must occur before the next action. Thus it is NOT immediate feedback to correct a student's paper of, say, ten problems immediately after she completes it. The student did not find out whether problem one was correct before starting problem two. In fact, in the learning stages, feedback for even smaller steps, such as parts of a single problem, often helps students learn faster than waiting for confirmation at the end of the entire problem. Good teachers follow such a principle already. They do not wait until the end of a class or even of a single explanation to nod or to inform a student of the quality of his oral contribution. The teacher immediately confirms a student's correct answer. Immediacy of consequences, then, is an aspect of the selection mechanism that an instructional programmer designs into every type of programmed instruction.

Activity Rate: High

Clearly, in order for consecutive factors to affect behavior, students must be active. Mere activity, however, is not enough. Activity must occur at a high rate. The technological expression of the science requires a high activity rate: the greater the frequency of responding, the more often the selective mechanism can work. Shaping occurs more quickly. Objectives are achieved more efficiently. Obviously, if there were no activity, then no shaping towards the desired repertoire could occur. As clear as such a requirement appears, designers of traditional instruction, especially computer-based instruction, deny, ignore, or overlook it. Most computer tutorials consist of screens filled with graphics or text, and student "interaction" consists of pressing a button to move onto a different screen (Cook, 1983; Vargas, J. S., 1986). Like traditional classroom lectures, such computer "tutorials" leave the student to decide what, of all the stuff presented, she should remember, and students must devise their own methods of learning. Again the basic paradigm differences come into play: Mainstream psychology encourages a model of teaching as "presenting information". It postulates that through internal structures and processes the student transforms the content of what he reads or hears into a useful performance. Behaviorology, in contrast, promotes a selectionist technology that requires behavior to be shaped. Programming requires overt activity at every instructional occasion: highly frequent and relevant actions provide an opportunity for postcedent events selectively to change those actions; changes that lead to forms of action not possible before.

Activity Rate: Relevant

But activity by itself is not enough. The student must respond to the appropriate stimuli–that is, he or she must get the right answer for the "right" reason. If the student can get answers by responding to aspects of a frame other than those the designer intended (for example, responding to grammatical forms instead of "content"), any advantage of overt responding over simply reading text is lost. In the 1960's and 1970's, Holland and his colleagues compared programmed instruction with the equivalent content presented in the usual text form. They found that students learned more from programmed instruction than from reading, but only when the words that students were asked to write were the main terms, definitions, principles being taught. Moreover, the amount of the printed material needed for responding was also important. When material that was not needed for responding was included (that is, students could respond correctly to the frames with that material blacked out) the effectiveness of the programming was correspondingly diminished. In a series of studies using this *blackout technique* Holland ranked programs by the percentage of material that could be blacked out without hindering responding. The ranking was inversely correlated to the superiority of the programs over reading text (Holland, 1967).

Successive Approximation

In building new repertoires, the instructional programmer, like a tutor, must start with steps the student can successfully perform, but must move as rapidly as possible towards the final goal. Each opportunity for students to respond provides an occasion for shaping behavior. The student response approximates behavior specified in the objectives. The approximation is successive: that is, the behavior changes only slightly in the direction of the designated final form.

In each frame, the student must do more, but he or she must be given enough help to master that more difficult step. Various ways to design these steps are described in the section on techniques of successive approximation, but general considerations follow based on the cybernetic character of programmed instruction.

Instructional steps that are too small don't stretch the student's repertoire, and they risk boring the student. At the same time, steps that are too large frustrate students who cannot perform correctly. No instructional designer can anticipate all of the ways students will react to a program. Instead, he or she prepares a first draft and gives it to students to try. Because it is easier to spot frames that ask too much (because students miss those items) than those that ask too little, it is a good idea to "overwrite" the first draft, using slightly larger steps than you think students will be able to do. Then, *sitting and watching* while each student works, you discover from errors, complaints, or the latencies to responding, where to insert extra frames. Traditionally when a student comes to an impasse, the designer does not explain, but writes an intermediate frame on the spot, passing it to the student with a comment such as "try this and see if it helps." It is important the designer not give help because, of course, the materials themselves must, in the end, teach on their own.

Watching a student work, as opposed to reviewing a record of responses later, also lets the instructional designer observe inappropriate stimulus control. One student in a tryout answered the third of a row of three items too fast to have read the material. When asked how he knew the answer so fast, he said, "Oh, there have been two "no's" so it had to be "yes". They never have three in a row." (Even when you tell students you have written a program, the students refer to the author as "they".) On another occasion, watching a student in a reading comprehension program, the author observed the student skipping the passage and going directly to the questions at the end, searching back for answers rather than reading through the passage. Since the point was to get students to read and remember what they read, a new format was adopted. Choices were embedded within the sentences until all readers began at the beginning. Both of these faults in stimulus control would have been missed had the designer looked only for items students missed, because the students responded correctly in both cases. And while a computer could have flagged the too short latency in the first student's response in a computer programmed instruction tryout, it would have missed the second. Only by watching students can you see where, in a frame, they are looking.

Two philosophies of programming exist. The first suggests testing and revising programs so that students only rarely make mistakes. Students are then carefully placed so that they take only programs they need, and for which they have prerequisite skills. Branching, in this approach, occurs in the sequence of lessons an individual takes. The second approach is to design a mainline sequence of instructional steps with branches within lessons to more detailed steps where students make mistakes. But branching should never be an excuse for lack of tryout. *It is NOT good programming to write a series of multiple-choice questions with explanations for why each wrong choice is wrong.* (For one thing it is punishing to go through a program getting item after item wrong.) Rather, branching should be used when students are likely to have different entry level skills for different parts of a program. For example, a health and safety program might have sections on eating habits, on exercising, and on drugs. Students might know about drugs, but not about the safe way to do situps. Branching would permit the student who correctly answers a test frame on drugs to skip that section. In the days of paper and pencil programs this format was called *gating.* A sample format would have a test frame at the top of each page. If you answered that item correctly you turned the page. If not, you worked down the page, ending with an item similar to the one you originally missed. With computer programs branching can be invisible. The student usually will not know whether he or she is in a branch.

A branching cybernetic program operates at many levels. The number of levels accommodates the variability range of student behavior so far encountered. Each expansion loop denotes a more detailed (deeper) level of program. Each deeper level (Levels II, III, and so on) is nested within the mainline program, Level I. Any given student need not encounter or be encumbered by the more detailed sequences of the deeper program levels. If an error is made at any frame, the student may be looped into a more detailed shaping sequence. But even without an error, if a frame carries with it a deeper level loop, the program provides the option for the student to select it if she wishes prior to responding to the frame before her.

Mastery Progression

The arrangement of frames, with their expansion loops consisting of more detailed frames, implies that student progression depends upon mastery. The student moves on to the next frame only after achieving success with the prior one. Depending upon the iterative status of the program–the number of times it has been reworked–the student may rarely experience failure. The program shapes a repertoire to be continually effective.

With programmed instruction, there is no need for a test in the usual sense of the word. A test is a sample of the behavior specified in behavioral objectives. But because students are performing throughout programmed instruction, the student is, in a sense, tested continually. While the programmer may identify quality control frames to review skills mastered at earlier parts of a program, no need exists for a quality control check at the end of the instructional program. Competency checks are built throughout a program. They constitute an integral part of programmed instruction.

A final word about student progression and mastery: The student progresses towards mastery defined not by how he or she performs in relation to other students, as in grading on a curve, but by what the instructional designer has defined as effective performance. All students thus finish a cybernetic program with an "A" level of performance. A general design feature of programmed instruction, then, is a 100 percent mastery of the basic skills it teaches. Students master all of the objectives, not just a portion of them. Further, they may master them not only by being accurate, but by being fluent, that is, capable of performing at a specified rate of endeavor established by their own baseline of performance.

SPECIFIC TECHNIQUES FOR SUCCESSIVE APPROXIMATION

The first frames in a sequence must be prompted. The student must be given the information needed to respond. From these initial frames, students progress through successive approximations, steps leading to the final performance. The steps should enable students to progress as fast as possible, but not so fast that they get difficult tasks before they can handle them. It takes considerable ingenuity to make the progression interesting. It is here that the instructional designer is artist as well as scientist.

Although only experience gives a feel for how exactly to break down skills into steps, there are some rules of thumb that may help. The following are four ways to increase difficulty. This following section draws heavily upon J. S. Vargas, *Behavioral psychology for teachers,* 1977, now being rewritten as *Educational behaviorology.*

Rule 1. Ask the student to select before asking him or her to produce. This sets the stage for shaping the composed response. The programmer must arrange for the student to emit simpler verbal behavior involving less effort. It is easier to respond to a correct answer than to recall it or to construct it. When you are teaching, all of the terms, abstractions, formulas, principles, and content (familiar to you), are likely to be new to your students. Students taking a program are like people introduced to several strangers at a party. It's not so easy to remember an unfamiliar name, but if someone says, "Was It Markle Q. Svezinski?" you can say, "Yes" or "No".

Almost a century ago, a teacher named Segun realized that "select" comes before "produce". He wrote three steps for teaching which are quoted in Maria Montessori's book, *The Montessori Method:*

> FIRST PERIOD. The association of the sensory perception with the name. For example, we present to the child, two colors, red and blue. Presenting the red, we say simply, "This is red," and presenting the blue, "This is blue." Then, we lay the spools upon the table under the eyes of the child.
> SECOND PERIOD. Recognition of the object corresponding to the name. We say to the child, "Give me the red," and then "Give me the blue."
> THIRD PERIOD. The remembering of the name corresponding to the object. We ask the child showing him the object, "What is this?" and he should respond, "Red." (Montessori, 1964, pp. 177-178)

The second period is a "select" step. "Give me the red" is a multiple-choice question with two choices, red and blue. In the third period, the student must "produce" the correct color name. Similarly, a student who cannot answer a question, such as "What is the name of a four-sided figure with only two parallel sides?" may pick out "trapezoid". Basically, the "select activity" defines an intermediate activity before moving to frames in which responses must be composed.

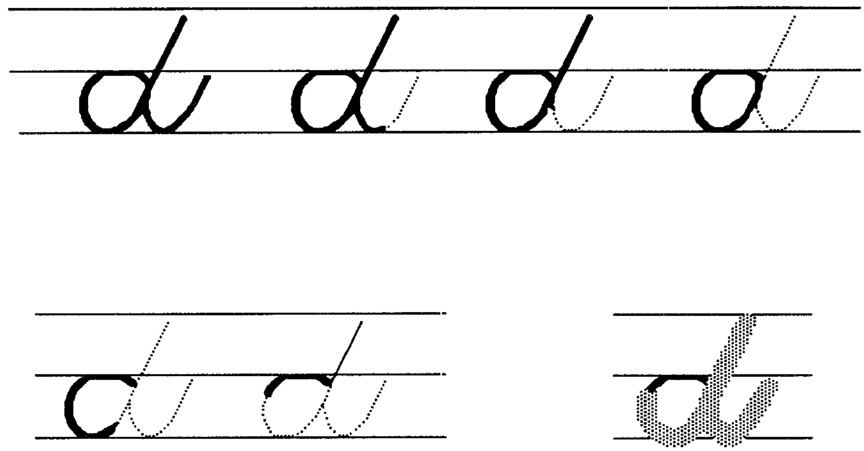

Fig. 1. Section of a program to teach handwriting. The child is asked to trace more and more of the letter, until he or she is writing the whole thing. Later exercises fade out the dots. The shaded part of the last example shows invisible ink which turns black when the student's yellow pen hits it. The tolerance for deviation from the sample can be made narrower as the program progresses.

Rule 2. Go from simple to complex and from part to whole. The student beginning to learn algebra will more successfully tackle a simple problem than a complex one and part of a large problem rather than all of it. The student who cannot pronounce a whole word may be able to say it by reading it syllable by syllable. We are proceeding from simple to complex when we add one-digit numbers before two-digit ones, drive in a quiet area before we drive downtown in rush hour, play "Pop Goes the Weasel" on the clarinet before "The Flight of the Bumble Bee", and analyze a simple poem before analyzing *Paradise Lost.*

We proceed from part to whole when we learn a swim-stroke by practicing using our hands and our breathing (standing waist deep in water) separately from our kick and then later combining them. Another example is "backward chaining", a technique in which the student finishes a partially completed problem, then more and more of the task until he or she does the whole unaided. Figure 1 shows part of a handwriting program which combined part-to-whole with fading (see below).

Rule 3. Start with prompts, and withdraw them as soon as possible. Suppose you were going to learn to fly an airplane. Using Rule 2 (simple to complex), you would probably start with a small single engine propeller plane (simple) rather than with a Boeing 747.

Your instructor would sit you in the left-hand seat in the cockpit, and after you were airborne the instructor would have you make some turns. Your instructor would probably have you use the steering wheel only, teaching you to use your feet to control the rudder later (Rule 2–part to whole). Other than that, there is no way to make the task simpler–you can't learn to turn one wing, then the other, then the whole plane. So the instructor demonstrates first, and now it's your turn. You turn the wheel left and the plane obediently tilts. Your instructor talks: "As soon as you are in the turn, straighten the wheel. The plane will continue by itself". You feel your wheel straighten out as the instructor works the dual control. The plane remains tilted and turning. "Apply a little back pressure. The plane will tend to descend in a turn, so you need to pull back". You pull back and the plane rises. "That's right. Not too much now", your instructor says. Now you are flying the airplane (making the whole response), but you are getting verbal help. Your instructor will continue to give you "crutches"—instructions and, if needed, physical help— until you can finally make the response unaided.

Probably the most common help that teachers first give and then gradually withdraw is verbal instructions. When teaching students to solve equations, it is helpful to "talk" them through the procedure first with auditory or printed instructions. In following the instructions, students' behavior is verbally-governed. But as they work the problems, their behavior becomes event-governed as they "get the feel" of solving, assembling, operating, or whatever the instructions are about.

In addition to instructions which are given only at first, prompts can be *faded* or *vanished.* For example, in teaching a student to label parts of a map or diagram, or to memorize a poem, the entire thing may be presented as a prompt (so that students merely need to copy), but then the "answers" fade, becoming increasingly difficult to see, or letters vanish a few at a time, so that only "hints" are left. You have probably used a similar technique when, for example, memorizing vocabulary in a foreign language. You covered the foreign words and tried to say them from the English equivalent, uncovering just as much as you needed to remember the entire word.

Rule 4. Go from easy to fine discriminations. Everything we learn involves discrimination. Selecting woods for a shop project, naming algae, or identifying composers, we learn to respond differently to different stimuli. The woodworker discriminates between woods, a biologist discriminates among algae, and the music historian discriminates between composers. Whenever we call something by a name, we are discriminating between it and similar objects.

What may be an obvious difference to an expert is rarely obvious to the beginner. The woodshop instructor picks out different woods as easily as most people pick out peas and beans. The biologist expert easily distinguishes among different algae; to the novice many look alike. The classical music enthusiast finds it difficult to understand how anyone could mistake Mozart for Beethoven. To others, Mozart and Beethoven sound the same.

In any field, some discriminations are more obvious than others. Students new to a woodworking program might have difficulty discriminating between types of wood. Faced with woods which have a very similar color and grain (such as spruce, white pine, and balsam), beginning students have a difficult task. But even a beginner can discriminate between redwood (which is reddish, as its name would suggest) and knotty pine (which is whitish with knots in it). By using the conspicuously different woods first, students can start off successfully; then more difficult discriminations can be added.

Inability to discriminate usually results from starting with too fine discriminations; even some supposedly "innate" handicaps ensue from inadequate shaping. People who call themselves tone deaf, for example, can tell the difference between a high note sung by a soprano and a low note sung by a man with a deep voice; that is a first approximation for telling pitches apart. By having the "tone deaf" student tell which note is higher and slowly decreasing the distance between tones–with, of course, immediate reinforcement for correct responding–better pitch discrimination can be taught. For the very young, the behaviorally delayed, or the physically handicapped, steps must be smaller and progress slower than for normal students, but anyone who can respond at all, can learn to discriminate better, if given easy enough discriminations to start with. Once students have learned to make the more obvious discriminations, they can progress to increasingly fine discriminations.

SUMMARY

Programmed instruction departs radically from the traditional view of pedagogy. Teaching is seen, not as the transmission of information, but as the shaping of student repertoires. It eschews the model of teacher as presenter, transmitting information that the student then absorbs. In fact, there is no such thing as "information". True, there are products of verbal behavior such as printed materials, or audio tapes. But even transmission advocates recognize that memorization of verbal products falls far short of the goals of education. In emphasizing goals such as "understanding", educators make clear that more must be "transmitted" than the form of the verbal material. True, students can learn by sitting in a lecture, but they will do so only if they are responding in ways called "paying attention". (Lecturers spend considerable effort on providing consequences for paying attention such as interesting demonstrations or humor to keep students listening instead of carrying on side conversations.) All educators are interested in shaping effective student behavior. But the fact that a student may repeat what an instructor says does not mean that "knowledge" has been "transmitted". Rather, text and instructions have been effective prompts, a part of the milieu that shaped student behavior.

Seeing the learning process as one in which behavior evolves throws the emphasis on the controls over student and teacher behavior. Objectives are written behaviorally, in terms of what students should be able to do. Instruction is broken down into steps for students to take–and care is taken that they respond to the relevant features of tasks or problems. Programs are revised according to how students respond. The educational setting becomes an interactive and cybernetic one; each party changing the other, as well as their milieu, over time.

Our country needs a more effective instructional technology. Although programmed instruction is still in its infancy, it offers a powerful alternative to the presentation mode and transmission models that clearly have not worked well with many students in today's schools. Programmed instruction, when constructed according to the principles outlined in this paper, offers an effective teaching technique for the future.

REFERENCES

Benjamin, Jr., L. T. (1988). A history of teaching machines. *American Psychologist, 43,* 703-712.
Cohen, H. L., and Filipczak, J. (1971). *A new learning environment.* San Francisco: Jossey-Bass.
Cook, D. E. (1983). CBT's feet of clay: Questioning the information transmission model. *Data Training, 3,* (12), 12-17.
Greer, R. D. (1980). *Design for Music Learning.* New York: Teachers College Press.
Holland, J. G. (1967). A quantitative measure for programmed instruction. *American Educational Research Journal, 4,* 87-101.
Lumsdaine, A. A., & Glaser, R. (Eds.). (1960). *Teaching machines and programmed instruction: a source book.* Washington, D.C.: National Education Association.
Montessori, M. (1964). *The Montessori method.* New York: Schaken.
Skinner, B. F. (1957). *Verbal Behavior.* New York: Appleton-Century-Crofts.
Skinner, B. F. (1981, July 31). Selection by consequences. *Science, 213,* 501-504.
Skinner, B. F. (1986). The evolution of verbal behavior. *Journal of the Experimental Analysis of Behavior, 45,* 115-122.
Vargas, E. A. (1986). Intraverbal behavior. In P. N. Chase & L. J. Parrot (Eds.), *Psychological aspects of language.* Springfield, Illinois: Charles C. Thomas.
Vargas, E. A. (1991a). Behaviorology: Its paradigm. In W. Ishaq (Ed.), *Human Behavior in Today's World.* New York: Praeger.
Vargas, E. A. (1991b). Verbal behavior: A four-term contingency relation. In W. Ishaq (Ed.), *Human Behavior in Today's World.* New York: Praeger.

Vargas, E. A., & Vargas, J. S. (1991). In W. Woodward & L. Smith (Eds.), *Skinner and American Culture*. Chapter submitted for publication.
Vargas, J. S. (1977). *Behavioral Psychology for Teachers.* New York: Harper & Row.
Vargas, J. S. (1984). What are your exercises teaching? An analysis of stimulus control in instructional materials. In W. L. Heward, T. E. Heron, D. S. Hill & J. Trap-Porter (Eds.), *Focus on behavior analysis in education* (pp. 126-141). Columbus: Charles E. Merrill.
Vargas, J. S. (1986, June). Instructional design flaws in computer-assisted instruction. *Phi Delta Kappan,* pp. 738-744.
Weinstock, R. (1984, May). A Title One tale: High reading/math gains at low cost in Kansas City, Kansas. *Phi Delta Kappan,* pp. 632-634.