Journal of Educational Psychology
1984, Vol. 76, No. 6, 1051–1058

Effects of Computer Programming on Young Children's Cognition

Douglas H. Clements and Dominic F. Gullo
Kent State University

Computers will soon be an integral part of the classroom and home environment of children, yet there are unanswered questions concerning their effects on young children's cognition. Particularly salient are largely unsubstantiated claims concerning the cognitive benefits of computer programming. This study assessed the effects of learning computer programming on children's cognitive style (reflectivity, divergent thinking), metacognitive ability, cognitive development (operational competence, general cognitive measures), and ability to describe directions. Eighteen 6-year-old children were pretested to assess receptive vocabulary, impulsivity/reflectivity, and divergent-thinking abilities. The children were then randomly assigned to one of two treatments, computer programming or computer-assisted instruction (CAI), that lasted 12 weeks. Posttesting revealed that the programming group scored significantly higher on measures of reflectivity and on two measures of divergent thinking, whereas the CAI group showed no significant pre- to posttest differences. The programming group outperformed the CAI group on measures of metacognitive ability and ability to describe directions. No differences were found on measures of cognitive development.

The increasing acceptance of the critical necessity for children to become computer literate is leading to an increased prominence of computers in the home and school environment. Yet there are unanswered questions regarding the effects of computer use on children's thinking. The purpose of this study was to investigate the effects of computer programming on 6-year-old children's cognitive style, metacognitive abilities, cognitive development, and ability to describe directions.

Seymour Papert, one of the creators of the computer language Logo and a leading exponent of the use of computer programming to expand children's intellectual power, based his ideas on the theories of Piaget, with whom he studied. Papert (1980) has argued that the most beneficial learning is what he calls "Piagetian learning," or "learning without being taught." He has proposed that computer programming environments can create conditions under which intellectual models take root, conditions in which young children can master notions formerly thought too abstract for their developmental level. Computers can make the abstract concrete and personal as they help children learn more effectively by making their thinking processes conscious. By programming the computer to do what they want it to do, children must reflect on how one might do the task oneself, and therefore, on how they themselves think (Papert, 1980). The computer programming environment holds the promise of being an effective device for cognitive process instruction—teaching *how*, rather than *what*, to think (Lochhead & Clement, 1979).

Although this is a relatively new area, some research has been done. Pilot work by the developers of the Logo computer language and others (Papert, 1980; Papert, Watt, diSessa, & Weir, 1979) indicates that children can learn to program and seem to profit intellectually. However, most of this work has, of necessity, been exploratory in nature, and much of it has been conducted with older children (e.g., 12-year-olds). Gorman and Bourne (1983) reported that third-grade children who worked for 1 hour a week on Logo programming performed significantly better on a test of rule learning than did children with ½ hour a week of

Requests for reprints should be sent to Douglas H. Clements, College of Education, 401 White Hall, Kent State University, Kent, Ohio 44242.

programming experience. There is some evidence that programming can increase problem-solving ability (Billings, 1983; Milner, 1973; Soloway, Lochhead, & Clement, 1982; Statz, 1974). Other reports indicate that considerable variability in skill levels attained by individual children exists and that children's programming ability is often limited to specific contexts (Pea, Hawkins, & Sheingold, 1983).

The little controlled research that exists gives only general directions concerning the possible cognitive benefits of programming. However, a few exploratory hypotheses can be advanced:

1. In Logo programming, children invent, construct, and modify their own projects; therefore, Logo programming might facilitate divergent thinking.
2. Because Logo is designed to encourage children to reflect on how they think, programming should lead them to develop metacognitive abilities, especially the ability to realize when they do and do not understand instructions.
3. Similarly, Logo programming may develop reflectivity in children as they think about their errors and how to correct them.
4. If computer programming can allow children to master ideas formerly thought too abstract for their developmental level, it may accelerate cognitive development, including operational competence (Papert, 1980).
5. Finally, because Logo programming involves giving explicit spatial commands, it should increase children's ability to describe directions from their own and others' perspectives.

It is possible, of course, that any benefits derived from computer programming can be attributed to interactive experiences with computers, rather than to the programming activity per se. It would therefore be necessary to provide a control group with computer experience not involving computer programming. Such experience might consist of computer-assisted instruction (CAI). CAI, with its roots in programmed learning, has a strong connection to the behavioristic tradition. Emerging from three themes of learning theory—individualization, behavioral objectives, and educational technology (Baker, 1978)—many CAI programs employ the approach of programmed learning. Thus they share the following characteristics: (a) They store a sequenced series of experiences, often providing alternative learning paths for individuals; (b) they offer independent pacing for individuals; (c) they give subjects controlled, contingent reinforcement; and (d) they evaluate performance quickly and accurately to provide data on the degree of mastery. Some research has been conducted on the feasibility and efficacy of using CAI with young children. It has been found that computers can effectively teach beginning skills and concepts (Billings, 1983; Hungate, 1982; Swigger & Campbell, 1981).

This study investigated the effects of experiences in computer programming (Logo), compared to experiences in computer-assisted instruction, on 6-year-old children's cognitive style (reflectivity, divergent thinking), metacognitive ability, cognitive development (operational competence, general cognitive measures), and ability to describe directions. Because the focus was on the effects of programming experience on cognition, no measures of standard achievement were employed.

Method

Subjects

Subjects for the study were 18 first-grade children (mean age, 6 years 11 months) from a middle-class, midwestern school system. Children were randomly assigned to one of two treatment groups: computer-assisted instruction (CAI) and computer programming (Logo). By chance, each group contained 5 boys and 4 girls.

Procedure

Subjects were pretested to determine pretreatment level of receptive vocabulary, reflectivity, and divergent thinking, using the instruments described in the next section. The computer activities were then implemented in two 40-min sessions a week for 12 weeks. Children worked in groups of 2 or 3 with one of the researchers. Researchers worked with both the programming and the CAI groups so that posttest differences between the groups would not be an artifact of differential adult interaction. At the end of 12 weeks, posttests were administered to assess children's cognitive style (reflectivity and divergent thinking), metacognitive ability, cognitive development (operational competence and general cognitive measures), and ability to describe directions.

Instruments

The following areas of cognition were assessed by the instruments listed.

Receptive vocabulary. The Peabody Picture Vocabulary Test—Revised, Form L (PPVT—R; Dunn & Dunn, 1981) was administered as a pretest measure of receptive vocabulary (internal consistency reliability for the appropriate age, $r = .77$). Dunn and Dunn (1981) also state that the PPVT—R is a useful measure of one aspect of language and cognition that suggests the level of present functioning of the child.

Reflectivity–impulsivity. As a measure of reflectivity, the Matching Familiar Figures Test (MFFT; Kagan, Rossman, Day, Albert, & Phillips, 1964) was administered. In the MFFT, the child is presented with a picture of a familiar object together with an array of highly similar variants. Only one of the pictures in the second array is an exact duplicate of the first. The child's task is to select the variant that is identical to the standard. Two measures are reported: the time in seconds it takes the child to choose the first picture (latency time) and the number of errors the child makes before choosing the correct standard. There are 12 items altogether, and the latency score and error score provide a measure of reflective thinking. Alternate forms of the MFFT were used for pre- and posttesting. Reported test–retest reliabilities for latency have ranged from .58 to .96; for error, they ranged from .39 to .80 (Messer, 1976).

Divergent thinking. As a measure of divergent thinking, the Torrance Test of Creative Thinking—Figural Test (Torrance, 1972) was administered. The Torrance test measures the ability of the child to think divergently in a nonverbal mode. It measures this ability in four ways: fluency—how many original ideas the child had (test–retest reliability, $r = .71$); flexibility—how varied these ideas were one from the other ($r = .73$); originality—how original the ideas were when compared to a normative group ($r = .85$); and elaboration—how many details were added to the main idea ($r = .83$). Alternate forms of the Torrance test were used for pre- and posttesting.

Awareness of comprehension failure. Markman (1977) developed two tasks whose purpose it was to measure children's ability to monitor and evaluate their own cognitive processes (metacognition). Two tasks are presented that are incomprehensible to all subjects. Children are presented instructions on how to perform a task, but crucial information for executing the task is deleted. The question is whether children realize they do not understand. In the first task, eight alphabet cards are divided equally between the experimenter and the child. Directions provided to the child include each player laying out one card at a time, looking for the "special card." However, there is no mention of what the special card might be. The second task involves a similarly incomplete description of a magic trick. Ten questions are asked in an attempt to ascertain whether the child realizes his or her lack of complete comprehension. Once the subject verbalizes a relevant question or statement the procedure is terminated. Markman's (1977) criteria for relevancy were employed. Interrater agreement was 95%, and test–retest reliability was .73. The questions provide increasingly specific prompts; for example, Question 1 is "That's it. Those are my instructions"; Question 2 is "What do you think?"; Question 9 is "Did I forget to tell you anything?" The children's score is the number of the question at which they indicate they realize that they do not understand; scores range from 1 to 11 (11 indicates the child never asked a question). Thus a lower number represents higher ability to monitor one's comprehension.

Operational competence (classification and seriation). To assess children's abilities in the logical operations of classification and seriation, four tasks were presented to them. These tasks were based on procedures developed by Inhelder and Piaget (1969) and are described in Clements (1983a). The classification tasks included (a) free classification (four items—sorting and resorting geometric shapes and familiar objects consistently (selecting three or more objects sharing logical attributes) and exhaustively (grouping all objects that possess an attribute) and (b) class inclusion (four items)—identifying the superordinate set as more numerous than the larger subordinate set (internal consistency reliability, $r = .75$). The seriation tasks included (a) seriation (five items)—ordering a series of objects by length, and (b) insertion (four items)—inserting a new object into an existing series ($r = .81$). The items were added for a possible test score of 8 for classification and 9 for seriation.

General cognitive measures. Four subtests of the McCarthy Screening Test (MST; McCarthy, 1978) were used as measures of cognitive development. The MST is a norm-referenced instrument adapted from the McCarthy Scales of Children's Abilities (McCarthy, 1972) whose purpose it is to assess children's performance on several cognitive measures known to be importantly related to school functioning. The four subtests were (a) right–left orientation (internal consistency reliability, $r = .32$); (b) verbal memory ($r = .54$); (c) draw-a-design ($r = .67$); and (d) numerical memory ($r = .69$).

Describing directions. To determine children's ability to accurately describe directions, they were administered a map test. They were presented with a simple street map and requested to draw a path from "home" at the upper right corner to the "store" at the lower left. They were then asked to describe their path for a friend who did not know the way (e.g., "Go straight three blocks. Then turn right . . ."). Their score was the percentage of correct verbal directions given. Test–retest reliability was .91.

Treatments

A sequence of training sessions in either (a) Logo (Terrapin) programming or (b) CAI was presented to the respective groups on four Apple II computers. The CAI treatment consisted of a sequenced set of computer-based lessons designed to teach aspects of the reading and arithmetic curriculum designated for the students. The sequence of lessons possessed the following characteristics: (a) provision of alternative learning paths for individuals progressing through a sequenced curriculum; (b) independent pacing for individuals; (c) provision of controlled, contingent reinforcement; and (d) provision of data on the degree of mastery. Commercial computer programs constituted the basis of the CAI treatment. They included (publisher and number of sessions are in parentheses) *Bumble Games* (The Learning Company; two); *Counting Bee* (EduWare; two); *Spelling Bee* (EduWare; two); *The Math Machine* (SouthWest EdPsych; four); *Elementary Math* (MECC; two); *Primary Math* (MECC; two); *Alphabetization* (Milliken; four); *Letter Recognition* (Hartley; two); and *The Reading Machine* (SouthWest EdPysch; four). These programs were chosen because they were designed to provide instruction and/or practice in the skills and abilities of the school system's mathematics, language arts, and reading curriculum. The sequences of lessons were constructed through a process of matching the objectives of the school's curriculum to the computer programs designed to teach these objectives. Each teacher worked with a group of 2 or 3 children on one computer, leading them in occasional discussions of the programs, their errors, and their corrections. Through guided questioning (e.g., "Why was that wrong? What will you do to fix it?"), children were asked to make their thinking and mastery of concepts explicit.

Following developmental principles applied to early childhood education, the programming experiences in the Logo computer language emphasized putting the children in control of the computer. Logo is a computer language designed to be accessible to children. It is developmentally appropriate for young children and is interactive. Children draw by directing the movements of a graphic "turtle," a small triangular pointer that can move around the display screen, leaving traces of its path (lines on the computer screen), in response to messages sent it by the programmer. Initial messages usually include directions for turning right or left and moving forward and backward. Logo is procedural—problems can be divided into small pieces, and a separate procedure written for each piece. In this way children can "divide and conquer" problems as they begin to see, in a concrete fashion, how tasks can be broken down into procedures, how procedures can be combined to form superprocedures, and how procedures interact. This illustrates that Logo is extensible—children can "teach" the computer new words. Following is an example of the Logo computer language, the commands it utilizes, and its struture, embedded in a condensed illustrative lesson from the computer programming treatment.

Children first planned what they would program the turtle to draw. They drew their pictures with black markers on a piece of paper. Two children drew a house and a flower. They then were encouraged to lay tracing paper over that picture and trace *one* piece of their drawing (e.g., the square that was the bottom of the house), labeling that "1." They used a clean sheet of tracing paper to trace another piece (the triangular roof), labeling that "2," and so on. These pieces were then defined as procedures that were typed into the computer with one of two special support programs (written by one of the researchers) that allowed the children to define a procedure and simultaneously watch it being executed, editing at any time that was necessary (see Clements, 1983b). These procedures provide a powerful method of "debugging," or correcting mistakes in, one's thinking. If a procedure was not doing what the children had anticipated, they were encouraged to think it through: "What did you tell the turtle to do? What *did* it do? What did you *want* it to do? How could you change your procedure?" and so on. Finally, they assembled the procedures in a superprocedure they might name HOUSE, which uses the SQUARE procedure to draw a square, moves into position, and uses the TRIANGLE procedure to draw a triangular roof. Each group of 2 or 3 children worked on one computer.

Children typed messages to the turtle at two levels, each of which involved the use of support programs. For the first 5 weeks children used a program that allowed them to direct the turtle by pressing single keys; for example, pressing F instructed the turtle to move forward 10 units; pressing L instructed the turtle to turn 30° to the left. Also, procedures were defined and executed with single keystrokes. For the last 7 weeks children typed in full Logo commands; for example, FORWARD 10 to move forward 10 units and LEFT 45 to turn 45° to the left. However, they still used a support program that allowed them to define a procedure while seeing each command executed immediately. This use of the full Logo language represented a step away from the intuitive and manipulative and toward the abstract and cognitive. It is more difficult but allows more flexibility and control.

The lessons were sequenced; children learned to move the turtle and draw lines using the single keystroke procedure (two sessions); write procedures to draw shapes with single keystrokes (four sessions); combine these procedures into superprocedures that draw more complex pictures (four sessions); write and combine procedures in the full language using the special support program described above (eight sessions); and plan the superprocedure first, then subdivide this into subprocedures ("top-down" planning; six sessions). Throughout, children were encouraged to "debug" their programs, locating and correcting errors.

Results

Table 1 presents the means and standard deviations of the Logo and CAI groups' scores on the various measures. To determine initial similarity between the CAI and Logo groups, a t test was performed on the means of the two PPVT—R scores. It revealed no significant difference between the groups, $t(16) = -.15$, *ns*.

As described in the Method section, the only two measures given at both pretest and posttest were measures of divergent thinking (Torrance Tests of Creative Thinking) and reflectivity–impulsivity (MFFT), both of which provide alternate test forms.

To determine if there were significant differences between pre- and posttest scores on the Torrance test for the CAI and Logo groups, a 2 (group) × 2 (pre–post) × 4 (sub-

Table 1
Means and Standard Deviations for Logo Programming and Computer-Assisted Instruction Groups

	Logo programming				Computer-assisted instruction			
	Pre		Post		Pre		Post	
Measure	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*
PPVT	111.00	8.47			111.67	10.58		
Torrance								
Fluency	46.00	7.28	58.89	10.64	38.66	7.02	41.22	7.63
Flexibility	48.44	7.59	53.11	13.58	41.78	7.58	41.66	8.41
Originality	51.67	13.73	66.56	16.80	48.11	17.38	49.67	13.48
Elaboration	40.44	10.37	50.00	15.77	33.11	4.65	37.00	7.50
Total	168.56	28.69	228.56	50.88	161.67	33.06	169.56	31.96
MFFT								
Error	16.56	5.70	7.67	4.09	14.11	4.78	11.88	5.44
Latency	6.94	2.26	14.31	10.75	10.41	3.66	10.06	7.92
Classification			6.67	1.40			5.00	1.56
Seriation			8.44	1.06			7.89	0.99
Metacognition								
Game task			4.33	2.96			9.11	2.76
Magic task			1.78	0.97			6.00	4.30
Describing directions			93.11	8.32			64.78	17.31
MST								
Left/right orientation			9.56	2.01			8.67	1.23
Verbal memory			28.11	2.62			28.33	1.32
Draw-a-design			15.44	2.07			14.67	3.00
Numerical memory (forward)			7.67	0.50			7.89	1.45
Numerical memory (reverse)			3.78	1.09			3.78	1.20

Note. PPVT = Peabody Picture Vocabulary Test; Torrance = Torrance Test of Creative Thinking—Figural Test; MFFT = Matching Familiar Figures Test; MST = McCarthy Screening Test.

test) analysis of variance was performed. Results of the analysis revealed significant main effects for group, $F(1, 48) = 7.44, p < .05$; pre–post, $F(1, 48) = 8.83, p < .01$; and subtest, $F(1, 48) = 19.36, p < .01$. A significant Group × Pre–Post interaction was also found, $F(1, 48) = 4.13, p < .05$. To find where the significant differences occurred in the interaction, post hoc analyses (Scheffé) were performed. For the Logo group there were significant differences from pretest to posttest on the following subtests: fluency, $p < .01$; originality, $p < .05$; and the overall divergent thinking score, $p < .01$.

To compare the Logo and CAI groups' pre- and posttest scores on the MFFT, correlated t tests were performed between the pre- and posttest means for latency and error scores. For the Logo group, analyses revealed a significant difference between the pre- and posttest scores for error, $t(8) = 3.58, p < .007$, and latency, $t(8) = 2.21, p < .05$. Table 1 shows that for the Logo group, the number of errors made was significantly reduced on the posttest and the latency time was significantly increased. Analyses showed no significant differences between pre- and posttest measures of error or latency for the CAI group.

Because there were no alternate forms available for the remaining measures and because the groups scored so similarly on all the pretests given, the Logo and CAI groups' scores on the remaining measures were compared using t tests following the treatment period. Analyses revealed that the Logo group required significantly fewer probe questions on the two metacognitive tasks than did the CAI group: game task, $t(16) = -3.54, p < .01$; magic task, $t(16) = -2.87, p < .02$. On the describing directions test, the number of directions given ranged from 4 to 15 for the Logo group ($M = 8.5$); for the CAI group, the number ranged from 5 to 9 ($M = 7.7$). The Logo group gave a significantly higher percentage of correct descriptions than did the CAI group, $t(16) = 4.42$, $p < .001$. That is, the Logo group provided more accurate verbal descriptions of the paths they had constructed. No significant differences were found between the Logo and CAI groups on any of the remaining measures.

Discussion

This study investigated the effects of computer programming in Logo on children's cognitive style (reflectivity, divergent thinking), metacognitive ability, operational competence, and overall cognitive development. The Logo and CAI (control) groups did not differ significantly prior to treatment in the language-cognitive domain as measured by the PPVT—R.

Analyses revealed significant pre- to posttest differences on the Torrance Tests of Creative Thinking for the Logo group on fluency and originality as well as on the overall divergent thinking score, while no significant differences were found for the CAI group. Children in the Logo group may have increased their ability to produce original ideas and to produce creative ideas as compared to a normative group because the Logo programming facilitated divergent thinking within a figural context. Cliatt, Shaw, and Sherwood (1980) found increased scores on the verbal but not figural Torrance tests, probably owing to the treatment's emphasis on verbal thinking. The emphasis on figural thinking in the present study's Logo treatment may have accounted for the increase on the fluency and originality subtest scores on the Torrance figural test. Because the CAI treatment often emphasized convergent rather than divergent thinking (e.g., timed presentations of addition exercises), it was expected that pre- to posttest differences within that group would not be significant.

Comparisons of pre- to posttest scores on the MFFT revealed significant differences for the Logo group on error and latency, while no significant differences were found for the CAI group. For the Logo group, the latency time increased and the number of errors decreased. It might be argued that the Logo treatment affected the number of errors directly, through the development of visual discrimination ability in the context of graphic programming. However, the CAI group also worked with programs that specifically taught visual discrimination, yet no pre- to posttest difference in the number of errors was found. The increase in latency time for the Logo group probably accounted for the decrease in errors on the MFFT, because children took more time to compare the pictures before choosing. The nature of programming in Logo necessitates thoughtful advanced planning, reflection on one's thinking, and explicit analysis of errors in "debugging," all of which may have accounted for the increase in latency time. The CAI treatment may not have affected latency time because it included instructional programs that rewarded quick responses.

The Logo programming group significantly outperformed the CAI group on both metacognition tasks. The ability to monitor one's own thinking and realize when one does not understand may also be positively affected by computer programming environments in which problems and solution processes are brought to an explicit level of awareness and in which consequent modification of problem solutions is emphasized. Through consistent feedback in the form of a visual representation of the procedures and sequences of their own thinking processes, children may have learned how to monitor those processes.

The scores on the test of describing directions were similarly affected. In the Logo group, children were required to visualize a graphic display and encode it into verbal directions from the turtle's perspective (the children were face to face with the turtle on the monitor). In this manner, the children practiced orienting to the turtle's visual perspective. This skill is a prerequisite to successful completion of the describing directions test, especially considering that the test was constructed such that directions involved left-to-right and top-to-bottom reversals. Research has shown that with practice children improve on visual perspective taking (Flavell, 1977; Hughes, cited in Donaldson, 1978).

No differences existed between the groups in two areas of cognitive development—operational competence (classification and seriation) and other specific aspects of cognitive development as measured by the MST. Thus, there was no evidence that 12 weeks of programming experience affects cognitive development compared to 12 weeks of CAI experience. One reason might be that while Papert based the Logo programming theory on the ideas of Piaget, this theory has not addressed certain Piagetian hypotheses that would tend to argue against the notion of the revolutionary potential of the computer—for example, that not even favorable environmental interventions can allow young children to deal with abstract concepts before they reach the period of formal operations (Piaget, 1964).

This study extends the work of others in demonstrating that programming can increase some aspects of problem-solving ability. Evidence is provided that programming may affect cognitive style; however, there is no evidence from this study that it affects general cognitive development. Future research needs to address questions from a developmental standpoint. It should be noted that the extensive computer training required for the treatments, along with the limited knowledge base we have regarding how to teach programming to such young children, necessitated the small sample size in the present investigation. The limitation of performing several t tests with a small number of subjects must be noted. As the knowledge base regarding appropriate techniques for teaching programming to very young children expands, future research should replicate this study using larger sample sizes.

References

Baker, F. B. (1978). *Computer managed instruction: Theory and practice.* Englewood Cliffs, NJ: Educational Technology Publications.

Billings, K. (1983). Research on school computing. In M. T. Grady & J. D. Gawronski (Eds.), *Computers in curriculum and instruction* (pp. 12–18). Alexandria, VA: Association for Supervision and Curriculum Development.

Clements, D. H. (1983a). A comparison: The effects of a logical foundations vs. a number skills curriculum on young children's learning of number and logical operations (Doctoral dissertation, State University of New York at Buffalo, 1983). *Dissertation Abstracts International, 44,* 94A. (University Microfilms No. 83-12, 401).

Clements, D. H. (1983b). Supporting young children's Logo programming. *The Computing Teacher, 11*(5), 24–30.

Cliatt, M. J. P., Shaw, J. M., & Sherwood, J. M. (1980). Effects of training on the divergent-thinking abilities of kindergarten children. *Child Development, 51,* 1061–1064.

Donaldson, M. (Ed.). (1978). *Children's minds* (pp. 13–16). Glasgow, Scotland: Fortang.

Dunn, L. M., & Dunn, L. M. (1981). *Peabody Picture Vocabulary Test—Revised.* Circle Pines, MN: American Guidance Service.

Flavell, J. (1977). The development of knowledge about visual perception. *Nebraska Symposium on Motivation, 25,* 43–76.

Gorman, H., Jr., & Bourne, L. E., Jr. (1983). Learning to think by learning LOGO: Rule learning in third-grade computer programmers. *Bulletin of the Psychonomic Society, 21,*165–167.

Hungate, H. (1982). Computers in the kindergarten. *The Computing Teacher, 9,*(5), 15–18.

Inhelder, B., & Piaget, J. (1969). *The early growth of logic in the child: Classification and seriation* (E. A. Lunzer & D. Papert, Trans.). New York: Norton.

Kagan, J., Rossman, B. L., Day, D., Albert, J., & Phillips, W. (1964). Information processing in the child. Significance of analytic and reflective attitudes. *Psychological Monographs, 78* (1, Serial No. 578).

Lochhead, J., & Clement, J. (Eds.). (1979). *Cognitive process instruction: Research on teaching thinking skills.* Philadelphia: Franklin Institute Press.

Markman, E. M. (1977). Realizing that you don't understand: A preliminary investigation. *Child Development, 48,* 986–992.

McCarthy, D. (1972). *McCarthy Scales of Children's Abilities.* New York: Psychological Corporation.

McCarthy, D. (1978). *McCarthy Screening Test.* New York: Psychological Corporation.

Messer, S. B. (1976). Reflection impulsivity: A review. *Psychological Bulletin, 83,* 1026–1052.

Milner, S. (1973, February). *The effects of computer programming on performance in mathematics.* Paper presented at the annual meeting of the American Education Research Association, New Orleans, LA. (ERIC Document Reproduction Service No. ED 076 391)

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* New York: Basic Books.

Papert, S., Watt, D., diSessa, A., & Weir, S. (1979). *Final report of the Brookline Logo Project: Project summary and data analysis* (Logo Memo 53). Cambridge, MA: Massachusetts Institute of Technology, MIT Logo Group.

Pea, R. D., Hawkins, J., & Sheingold, K. (1983, April). *Developmental studies on learning Logo computer programming.* Paper presented at the Biennial Meeting of the Society for Research in Child Development, Detroit, MI.

Piaget, J. (1964). Development and learning. In R. E. Ripple & V. N. Rockcastle (Eds.), *Piaget rediscovered.* Ithaca, NY: Cornell University Press.

Soloway, E., Lochhead, J., & Clement, J. (1982). Does computer programming enhance problem solving ability? Some positive evidence on algebra word problems. In R. Seidel, R. Anderson, & B. Hunter (Eds.), *Computer literacy* (pp. 171–185). New York: Academic Press.

Statz, J. (1974). The development of computer programming concepts and problem solving abilities among ten-year-olds learning Logo (Doctoral dissertation, Syracuse University, 1973). *Dissertation Abstracts International, 34,* 5418B–5419B. (University Microfilms No. 74-10,180)

Swigger, K., & Campbell, J. (1981). Computers and the nursery school. In *Proceedings of the National Educational Computing Conference* (pp. 264–267). Iowa City, IA: National Educational Computing Conference.

Torrance, E. P. (1972). *Torrance Tests of Creative Thinking: Figural Test.* Lexington, MA: Personnel Press.

Received December 12, 1983
Revision received February 17, 1984 ■

Correction to Leventhal et al.

The article "Primacy/Recency Effects in Student Ratings of Instruction: A Reinterpretation of Gain–Loss Effects" by Les Leventhal, Shelly J. C. Turcotte, Philip C. Abrami, and Raymond Perry (*Journal of Educational Psychology*, 1983, Vol. 75, No. 5, pp. 692–704) requires corrections. The affiliation of Philip Abrami should be Concordia University. On page 694, in the discussion of the relation between gain–loss findings and primacy/recency effects, the third sentence in the second paragraph should read, "One need only suppose that (a) negative primacy and positive recency effects have occurred and (b) the absolute size of the recency effect was larger than the absolute size of the primacy [not primary] effect." On page 695, in the *Present Experiment* section, the penultimate sentence in the left-hand column should read, "We directed students to evaluate only Lecture 2—instead of saying nothing about which portion of the 'course' was to be evaluated—in order to determine whether primacy effects would occur when instructions were written to prevent them." On page 701, in the note to Table 4, Footnote a should read "Difference between 2.54 and 2.22 was not significant."