

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing [scholarworks-group@umbc.edu](mailto:scholarworks-group@umbc.edu) and telling us what having access to this work means to you and why it's important to you. Thank you.

# Encyclopedia of Information Technology Curriculum Integration

Lawrence A. Tomei  
*Robert Morris University, USA*

Volume I  
A–Interactive Videoconferencing

Information Science  
**REFERENCE**

**INFORMATION SCIENCE REFERENCE**

Hershey • New York

Acquisitions Editor: Kristin Klinger  
Development Editor: Kristin Roth  
Senior Managing Editor: Jennifer Neidig  
Managing Editor: Sara Reed  
Assistant Managing Editor: Carole Coulson  
Copy Editor: Ashlee Kunkle, Jeanie Porter, Angela Thor  
Typesetter: Amanda Apicello, Larissa Vinci, Carole Coulson  
Cover Design: Lisa Tosheff  
Printed at: Yurchak Printing Inc.

Published in the United States of America by  
Information Science Reference (an imprint of IGI Global)  
701 E. Chocolate Avenue, Suite 200  
Hershey PA 17033  
Tel: 717-533-8845  
Fax: 717-533-8661  
E-mail: [cust@igi-global.com](mailto:cust@igi-global.com)  
Web site: <http://www.igi-global.com/reference>

and in the United Kingdom by  
Information Science Reference (an imprint of IGI Global)  
3 Henrietta Street  
Covent Garden  
London WC2E 8LU  
Tel: 44 20 7240 0856  
Fax: 44 20 7379 0609  
Web site: <http://www.eurospanonline.com>

Copyright © 2008 by IGI Global. All rights reserved. No part of this publication may be reproduced, stored or distributed in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher.

Product or company names used in this set are for identification purposes only. Inclusion of the names of the products or companies does not indicate a claim of ownership by IGI Global of the trademark or registered trademark.

IGI Global, [www.igi-global.com](http://www.igi-global.com). Posted by permission of the publisher.

British Cataloguing in Publication Data  
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this encyclopedia set is new, previously-unpublished material. The views expressed in this encyclopedia set are those of the authors, but not necessarily of the publisher.

*If a library purchased a print copy of this publication, please go to <http://www.igi-global.com/reference/assets/IGR-eAccess-agreement.pdf> for information on activating the library's complimentary electronic access to this publication.*

# Behavior Analysis and ICT Education: Teaching Java™ with Programmed Instruction and Inter-teaching

**Henry H. Emurian**

*Univeristy of Maryland – Baltimore County, USA*

## INTRODUCTION

Acquiring skill in computer programming is acknowledged to be valuable for information science students (Forgionne, 1991). Educators in the discipline, however, recognize that students may sometimes select management information systems (MIS) and related academic majors to avoid the programming demands of a computer science curriculum (Gill & Holton, 2006). Although object-oriented software methodologies are included in undergraduate curriculum recommendations for information systems programs (e.g., IS 2002, presented in Gorgone et al., 2002) and information technology programs (e.g., IT 2005, presented in SIGITE, 2005), the complexity and instability of object-oriented languages such as Java<sup>1</sup> pose additional burdens on both students and educators alike (Roberts, 2004). Moreover, the diversity challenges of a typical freshman class in computer programming are highlighted by Koen (2005): “Freshman are very diverse with respect to their entering computer skills—some are state computer champions, while others have never touched a computer before” (p. 599). Realizing these challenges and given a course in Java that is intended to be taken by information systems majors, what instructional approach should the teacher adopt to maximize student learning?

Educators have struggled for decades to solve that problem. Instructional recommendations include support to understand logical constructions and flow of control (Papert, 1980), intelligent computer assisted instruction (Anderson & Skwarecki, 1986), approaches to classroom teaching and student learning (Mayer, 1988), emphasis on mathematics and algorithms (Hu, 2006), and other supportive programming environments such as BlueJ (Kolling, Quig, Patterson, & Rosenberg, 2003), DrJava (Hsia, Simpson, Smith, & Cartwright, 2005), Problem-Based Learning (Tsang & Chan, 2004),

and the Environment for Learning to Program (Truong, Bancroft, & Roe, 2005). It is not uncommon, moreover, for instructors to avoid responsibility for the outcome of their teaching and to hold the student solely accountable for any failure rather than concluding that the pedagogy might have been flawed (Jenkins, 2001).

The instructional approach taken at the University of Maryland – Baltimore County (UMBC), however, is similar to the intent of many of the above recommendations to assist new learners. Our aim is to expose novitiate students at the outset to a series of instructional events, as the first technical exercise in a Java course, that results in all students being able to write and to understand the JApplet program presented in Table 1, which will display a text string in a browser window on the Web.

The initial learning during the first class, which involves completion of a Web-based tutoring system, is supported by a subsequent lecture on the program during the second class, when the students run the program on the Web. A final elaboration and consolidation event takes place during the third class, when the students engage in dyadic collaborative peer tutoring to test each other’s knowledge and understanding of the program, to raise questions as needed, and to confirm each other’s mastery of the program within a social context. These intensive initial learning experiences are in furtherance of preparing the student to be taught with lecture, demonstrations, and peer collaboration throughout the remaining classes of a semester. The Java code that is mastered in a typical course will produce a final JApplet project that will run on the Web.<sup>2</sup>

From the perspective of teaching computer programming, these techniques together converge on what is increasingly recognized as vital ingredients to facilitate science education, in general (DeHaan, 2005). Among several recommendations of learning principles

Table 1. Each cell with Java code reflects a learn unit in the tutor

Line						
1	import	javax.swing.JApplet	;			
2	import	javax.swing.JLabel	;			
3	import	java.awt.Color	;			
4	public	class	MyProgram	extends	JApplet	{
5	JLabel	myLabel	;			
6	public	void	init()	{		
7	myLabel	=	new	JLabel("This is my first program.")	;	
8	getContentPane()	.	setBackground(Color.yellow)	;		
9	getContentPane()	.	add(myLabel)	;		
10	}					
11	}					

to promote retention and transfer of knowledge, for example, are repeated practice with different instructional modalities (Halpern & Hakel, 2003) and with socially supported interactions (Fox & Hackerman, 2003). The remaining sections of this article, then, present the intellectual context and the educational technology to implement behaviorally oriented instructional tactics as a solution to the general problem of effective pedagogy.

## BACKGROUND

The instructional tactics adopted in the classroom at the start of a semester's work are based upon *programmed instruction* (PI), which is a form of structured and optionally automated instruction, and *interteaching*, which is a form of collaborative peer tutoring. As implemented in the present context, these tactics originated from behavior analysis, and the Cambridge Center for Behavioral Studies<sup>3</sup> provides fundamental definitions and a wealth of information regarding the philosophical underpinnings and applications of this approach to science, in general, and education, in particular. The classroom applications under consideration are "atheoretical" in that the causes and explanations for the development of a complex repertoire of programming skill are assumed to rely in a series of systematically crafted interactions as the antecedents to knowledge and skill for the individual student. This orientation

contrasts with indirect and metaphorical explanations of behavior, such as intelligence, personality, locus of control, understanding, engaged academic time, and mental models (Emurian & Durham, 2003; Greer & McDonough, 1999).

A general treatment of behavior analysis applications to education, which includes consideration of programmed instruction and personalized collaborations, is presented in Greer (2002). Lockee, Moore, and Burton (2004) present a comprehensive summary of the literature related to the components of PI and to the context of its use, and Feurzeig (2006) provides a historical perspective of educational technology developments that commence with programmed instruction and that conclude with "intelligent" computer-aided instruction. This section will present an overview of the instructional technology associated with programmed instruction and interteaching, and a later section will present the applications to teach Java.

## Programmed Instruction

Programmed instruction is a technique to structure textual information in small units for the student to study and to master at the level of a unit. Each unit, which is referred to as a "frame," consists of text along with a test that provides the opportunity to demonstrate learning. The test could require the completion of a partially-spelled word in a sentence or it could require completing a statement by filling in a blank space

with a word or words. The frames are designed so that correct answers are required in one frame before a subsequent frame is presented, and the frames increase in complexity over the course of learning. The design features of programmed instruction, synthesized from the literature, are presented in Emurian (2005), and the intellectual history of this instructional technology, as it relates to behaviorally oriented computer-based tutoring systems, is discussed in Emurian and Durham (2003). The components of PI that are most relevant to the present discussion are as follows: (1) learner constructed responses based on recall, (2) immediate feedback for performance, (3) successive approximations to a final learning objective, and (4) learner-paced progress (Holland, 1960; Scriven, 1969; Skinner, 1958; Vargas & Vargas, 1991). An example of a programmed instruction textbook (Holland & Skinner, 1961) may be freely downloaded from the Cambridge Center for Behavioral Studies.<sup>4</sup>

The origin of programmed instruction is attributable to B.F. Skinner (1904-1990), a behavioral psychologist who spent most of his academic career at Harvard University. In a pioneering paper (Skinner, 1954), the argument was advanced that principles of learning derived from laboratory experimentation could be directly applied to the design of instruction that would manage the countless moment-to-moment interactions between a student and a teacher that might be essential for each and every learner to reach a similar criterion of mastery, which was the objective. Given the complexity and frequency of these interactions, machine support was proposed as a reasonable, if not absolutely essential, requirement for the successful implementation of such a teaching technology (Skinner, 1958). Advanced learners such as college students, however, might be anticipated to follow the process of learning when the instructional frames were presented in the form of a textbook (Holland & Skinner, 1961).

Programmed instruction was not widely adopted by educators, even following the advent of the computer as the “machine” that would implement the interactive information system. Among the reasons given to explain the paucity of PI applications are included the rigidity of the step-like frames and the perhaps questionable assumption that the process for the development of a complex verbal repertoire could never be captured in sufficient detail as to be programmable (McDonald, Yanchar, & Osguthorpe, 2005). Although the impact of automated tutoring systems based on other models

of learning is evident in such applications as the Cognitive Tutors, which have been designated as one of five exemplary curricula in K-12 mathematics education by the U.S. Department of Education (Mathan & Koedinger, 2005), behavior analysis of problem solving and similar “cognitive” phenomena has recently been undertaken within the context of computer-interactive learning of the rectangular coordinate system based on a matching-to-sample procedure (Ninness et al., 2005). Finally, an emerging relational-frame theory (Hayes, Barnes-Holmes, & Roche, 2001) shows promise to operationalize language acquisition and use in terms accessible to a behavior analysis.

The emergence of personal computers gave rise to a renewed interest in programmed instruction among researchers and practitioners. One of the first computer-based applications of PI to appear in the behavioral literature was reported by Dube, McDonald, McIlvane, and Mackay (1991). These investigators showed that a computer-based tutoring system, based on a matching-to-sample paradigm, could be used to manage the several developmental steps required to teach two mentally retarded adults to spell. More relevant to the management of text-based learning systems, Tudor and Bostow (1991) reported that a microcomputer PI system that taught PI design achieved the best learning outcome in college students when the frames required overt constructed responses during knowledge acquisition. These results were later confirmed and extended to show that a relatively high density of constructed responses during learning produced superior post-tutor test performance (Kritch & Bostow, 1998). Related research, based upon a microcomputer version of Holland and Skinner (1961), investigated parameters of the temporal transition between successive frames, with a delay of several seconds generally supporting improved response accuracy across the frames (Kelly & Crosbie, 1997). With the exception of the PI tutoring system to support the learning of Java, which will be described below, the most recent study in the behavioral literature that used programmed instruction applied the design philosophy to teach preschoolers how to point with the computer mouse (Shimizu & McDonough, 2006). Although many of these studies departed somewhat from the formalism of PI, all are in furtherance of showing the application of behavior principles to computer-based tutoring systems. Finally, as evidenced by the Java tutoring system developed by Truong et al. (2005), which requires constructed responses by a

learner, functional aspects of PI are present in effective tutors even when behavior analysis is not identified as a rationale for the system.

## Interteaching

Structured collaboration among students is increasingly recognized as a valuable experience to promote learning computer programming (Jehng, 1997; Williams, Wiebe, Yang, Kerzli, & Miller, 2002), to include Java (Beck, Chizhik, & McElroy, 2005). In behavior analysis, one collaborative paradigm is interteaching, defined by Boyce and Hineline (2002) as “a mutually probing, mutually informing conversation between two people” (p. 220). During an interteaching session, two students discuss their answers to questions that are presented in a study manual that is used to prepare for the session. The intent of the session is for both students to reach consensual agreement on the understanding of the material and to move beyond factual recitations into generalizable rules and concepts. The students submit a report of the effectiveness of the session, and the report may contain questions that should be resolved by an instructor. The knowledge gained from the interteaching session is assessed during regularly scheduled tests, which may include objective test items, short answer questions, and essays.

Interteaching is derived from the personalized system of instruction (PSI), which was developed by the behavioral psychologist Fred Keller (Keller, 1968). One aspect of the PSI is to use a student “knowledge expert” to determine, generally within an interpersonal interaction, that a student “knowledge novice” has mastered a unit of material and is prepared to advance to a subsequent unit in a course of study. Student learners may repeat a unit of study until it is passed, with no grade penalty, and progress is determined by the pace set by the learner. The behavioral contingencies are said to optimize learning in all students and to remove the aversive control or coercion that may be prevalent in other instructional techniques.

Research and classroom applications generally support the effectiveness of the “Keller method” and its superiority to lecture courses over a wide range of topics and students (Kulik, Kulik, & Cohen, 1979), and the method’s value in engineering education has been documented (Canelos & Ozbeki, 1983; Haws, 1998). As suggested in the historical and evaluative perspective by Grant and Spencer (2003), however,

the absence of widespread use of the method is likely attributable, paradoxically, to its effectiveness, which requires a considerable expenditure of energy by teachers. Importantly, the method requires an orientation to pedagogy that favors overcoming individual differences in student achievement rather than simply documenting differences (Emurian, 2001; Haws, 2000). As stated by Murray Sidman in an address to the Eastern Psychological Association,

*The PSI system of instruction is a behavior analytic contribution that has the potential, like successful public health measures, to exert population-wide effects, but behavior analysis has not yet come up with methods for gaining acceptance of that contribution by either the education establishment or by the general public.* (Sidman, 2006, p. 240)

The advent of the Internet may well help to foster just such an acceptance.

There is growing evidence that the PSI framework is being adopted for Web-based instructional applications. Martin, Pear, and Martin (2002) adopted a computer-aided PSI format for an undergraduate psychology course in which electronic interactions occurred between student “proctors” and student “learners.” Koen (2005) reported the use of Webcams and related synchronous and asynchronous communication media to implement some of the interpersonal proctoring and other social (i.e., “presence”) aspects for a Web-based PSI computer course for freshman, which included Java among other technical topics that were taught. Although behavior principles were not mentioned, Xu, Wang, and Wang (2005) reported a conceptual model of personalized virtual learning environments that convey many of the essential elements of the PSI, to include units of study that meet the level of the individual learner. As suggested by Folkers (2005), moreover, the accelerated movement of educational offerings from the physical “marketplace” to the virtual world of the “marketspace,” through the integration of distance education programs into curriculum models, may provide the occasion for those long sought population-wide effects as a consequence of the implementation and effectiveness of behavior analytic techniques.

## TEACHING JAVA WITH PROGRAMMED INSTRUCTION AND INTERTEACHING

For the past several years, the Information Systems Department at UMBC has adopted programmed instruction and interteaching as components of a Java course intended to be taken by information systems undergraduate and graduate students who have minimal programming experience and no prior experience with Java.<sup>5</sup> The PI tutoring system, which takes approximately 3 hours for a student to complete, is a Web-based system that leads the learner to an understanding of the code displayed within each cell of Table 1. When a student exits the tutor, the student has passed online multiple-choice tests on the items and lines of code (i.e., cells and rows), and the student has written the code from memory. This instructional experience is followed by a lecture and finally by an interteaching session, and that latter session is intended to provide the opportunity to solidify and elaborate the prior learning. Several assessments of learning and software self-efficacy are taken throughout these initial classes, as given below for a typical sequence of events.

- **Class 1:**
  1. Pretutor assessment
    - a. Java software self-efficacy
    - b. Multiple-choice tests on general programming principles
  2. Java PI tutoring system
  3. Post-tutor assessment
- **Class 2:**
  1. Lecture on the code while students enter the code into a UNIX™ text editor.
  2. Run the JApplet on the Web.
  3. Post-lecture assessment
- **Class 3:**
  1. Interteaching session on the Java code
  2. Post-interteaching assessment

The data presented in our reports show that students develop skill and confidence that emerge cumulatively and synergistically over the several learning experiences (Emurian, 2004, 2005, 2006a, 2006b; Emurian & Durham, 2003; Emurian, Wang, & Durham, 2003). The design of the PI tutoring system promotes meaningful learning (Mayer, 2002), as evidenced by the students' acquired competency to generalize knowledge gained from the tutor frames and constructed response

requirements. Students report value in and appreciation of all aspects of these events, and students who were apprehensive at the outset seem to benefit most from having an initially positive experience in a computer programming course, some for the very first time in their student careers. This sets the occasion for the students' continued involvement in the course with confidence and enthusiasm.

## CONCLUSION

Although educators might have the success of their students as a primary goal of teaching, it is less certain that what happens in the classroom is based on empirical evidence of effectiveness: a rational pedagogy. And it is sometimes the case that expecting students prematurely to solve general computer programming problems and to understand complex control structures and algorithms neglects the propaedeutic skills that students must possess to undertake such higher-order learning. In furtherance of providing those skills to our students, techniques derived from behavior analysis have been demonstrably effective in promoting skill, confidence, and meaningful learning by novitiate students regarding an object-oriented programming language. Perhaps overlooked in other models of automated instruction (Anderson, Corbett, Koedinger, & Pelletier, 1995), behavior analysis excels in identifying the ontogenetic instructional learn units (Greer & McDonough, 1999) whose mastery provides the expressive verbal tools for advanced understanding, thinking, and problem solving in the domain of computer programming and beyond (Skinner, 1957). With such a background providing the propaedeutic repertoire, students will progress to more advanced learning of object-oriented programming rules and concepts, with no student left behind.

## REFERENCES

- Anderson, J.R., Corbett, A.T., Koedinger, K.R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*, 4(2), 167-207.
- Anderson, J.R., & Skwarecki, E. (1986). The automated tutoring of introductory computer programming. *Communications of the ACM*, 29(9), 842-849.

- Beck, L.L., Chizhik, A.W., & McElroy, A.C. (2005). Cooperative learning techniques in CS1: Design and experimental evaluation. In *Proceedings of the Thirty-Sixth SIGCSE Technical Symposium on Computer Science Education* (SIGCSE 2005) (pp. 470-474).
- Boyce, T.E., & Hineline, P.N. (2002). Interteaching: A strategy for enhancing the user-friendliness of behavioral arrangements in the college classroom. *The Behavior Analyst*, 25, 215-226.
- Canelos, J., & Ozbeki, A. (1983). Application of the Keller instructional strategy of personalized instruction for the improvement of problem-solving learning in technical education. *Journal of Instructional Psychology*, 10(2), 61-69.
- DeHaan, R.L. (2005). The impending revolution in undergraduate science education. *Journal of Science Education and Technology*, 14(2), 253-269.
- Dube, W.V., McDonald, S.J., McIlvane, W.J., & Mackay, H.A. (1991). Constructed-responses matching to sample and spelling instruction. *Journal of Applied Behavior Analysis*, 24(2), 305-317.
- Emurian, H.H. (2001, April-June). The consequences of e-Learning (Editorial). *Information Resources Management Journal*, 14(2), 3-5.
- Emurian, H.H. (2004). A programmed instruction tutoring system for Java™: Consideration of learning performance and software self-efficacy. *Computers in Human Behavior*, 20(3), 423-459.
- Emurian, H.H. (2005). Web-based programmed instruction: Evidence of rule-governed learning. *Computers in Human Behavior*, 21(6), 893-915.
- Emurian, H.H. (2006a). A Web-based tutor for Java™: Evidence of meaningful learning. *Journal of Distance Education Technologies*, 4(2), 10-30.
- Emurian, H.H. (2006b). Assessing the effectiveness of programmed instruction and collaborative peer tutoring in teaching Java™. *International Journal of Information and Communication Technology Education*, 2(2), 1-16.
- Emurian, H.H., & Durham, A.G. (2003). Computer-based tutoring systems: A behavioral approach. In J.A. Jacko & A. Sears (Eds.), *Handbook of human-computer interaction* (pp. 677-697). Mahwah, NJ: Lawrence Erlbaum & Associates.
- Emurian, H.H., Wang, J., & Durham, A.G. (2003). Analysis of learner performance on a tutoring system for Java. In T. McGill (Ed.), *Current issues in IT education* (pp. 46-76). Hershey, PA: IIR Press.
- Feurzeig, W. (2006). Educational technology at BBN. *IEEE Annals of the History of Computing*, 28(1), 18-31.
- Folkers, D.A. (2005). Competing in the marketplace: Incorporating online education into higher education – an organizational perspective. *Information Resources Management Journal*, 18(1), 61-77.
- Forgionne, G.A. (1991). Providing complete and integrated information science education. *Information Processing & Management*, 27(5), 575-590.
- Fox, M.A., & Hackerman, N. (2003). *Evaluating and improving undergraduate teaching in science, technology, engineering, and mathematics*. Washington, DC: The National Academies of Science Press.
- Gill, T.G., & Holton, C.F. (2006). A self-paced introductory programming course. *Journal of Information Technology Education*, 5, 95-105.
- Gorgone, J.T., Davis, G.B., Valacich, J.S., Topi, H., Feinstein, D.L., & Longenecker, H.E. (2002). IS 2002. Model curriculum and guidelines for undergraduate degree programs in information systems. Retrieved September 27, 2007, from <http://www.aisnet.org/Curriculum/IS2002-12-31.pdf>
- Grant, L.K., & Spencer, R.E. (2003). The personalized system of instruction: Review and applications to distance education. *International Review of Research in Open and Distance Learning*, 4(2), 1-22.
- Greer, R.D. (2002). *Designing teaching strategies: An applied behavior analysis systems approach*. New York: Academic Press.
- Greer, R.D., & McDonough, S.H. (1999). Is the learn unit a fundamental measure of pedagogy? *The Behavior Analyst*, 22(1), 5-16.
- Halpern, D.F., & Hakel, M.F. (2003). Applying the science of learning to the university and beyond: Teaching for long-term retention and transfer. *Change*, 35(4), 37-41.
- Haws, D.R. (1998, November). Personal reflections on PSI in engineering mechanics. In *Proceedings of the*

*Frontiers in Education National Conference*, Tempe, Arizona (pp. 280-285).

Haws, D.R. (2000, June). Teacher gone: The marginalization of PSI in engineering education. In *Proceedings of the American Society for Engineering Education*, St. Louis, MO.

Hayes, S.C., Barnes-Holmes, D., & Roche, B. (2001). *Relational frame theory: A post-Skinnerian account of human language and cognition*. New York: Kluwer Academic/Plenum Publishers.

Holland, J.G. (1960). Teaching machines: An application of principles from the laboratory. *Journal of the Experimental Analysis of Behavior*, 3, 275-287.

Holland, J.G., & Skinner, B.F. (1961). *The analysis of behavior: A program for self-instruction*. New York: McGraw-Hill Co.

Hsia, J.I., Simpson, E., Smith, D., & Cartwright, R. (2005, February 23-27). Taming Java for the classroom. In *SIGCSE '05*, St. Louis, MO (pp. 327-331).

Hu, C. (2006). It's mathematical after all: The nature of learning computer programming. *Education and Information Technologies*, 11(1), 83-92.

Jehng, J.-C. J. (1997). The psycho-social processes and cognitive effects of peer-based collaborative interactions with computers. *Journal of Educational Computing Research*, 17(1), 19-46.

Jenkins, T. (2001). Teaching programming – a journey from teacher to motivator. In *2nd Annual LTSN-ICS Conference*, London. Retrieved September 27, 2007, from <http://www.ics.ltsn.ac.uk/pub/conf2001/papers/Jenkins.htm>

Keller, F.S. (1968). Goodbye teacher. *Journal of Applied Behavior Analysis*, 1, 79-89.

Kelly, G., & Crosbie, J. (1997). Immediate and delayed effects of imposed postfeedback delays in computerized programmed instruction. *The Psychological Record*, 47, 687-698.

Koen, B.V. (2005). Creating a sense of “presence” in a Web-based PSI course: The search for Mark Hopkins’ log in a digital log. *IEEE Transactions on Education*, 48(4), 599-604.

Kolling, M., Quig, B., Patterson, A., & Rosenberg, J. (2003). The BlueJ system and its pedagogy. *Journal of Computer Science Education*, 14(Dec), 1-12.

Kritch, K.M., & Bostow, D.E. (1998). Degree of constructed-response interaction in computer-based programmed instruction. *Journal of Applied Behavior Analysis*, 31, 387-398.

Kulik, J.A., Kulik, C.C., & Cohen, P.A. (1979). A meta-analysis of outcome studies of Keller’s personalized system of instruction. *American Psychologist*, 34, 307-318.

Lockee, B., Moore, D.M., & Burton, J. (2004). Foundations of programmed instruction. In D.H. Jonassen & P. Harris (Eds.), *Handbook of research on educational communications and technology* (pp. 545-570). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Martin, T.L., Pear, J.J., & Martin, G.L. (2002). Analysis of proctor marking accuracy in a computer-aided personalized system of instruction course. *Journal of Applied Behavior Analysis*, 35(3), 309-312.

Mathan, S.A., & Koedinger, K.R. (2005). Fostering the intelligent novice: Learning from errors with metacognitive tutoring. *Educational Psychologist*, 40(4), 257-265.

Mayer, R.E. (1988). *Teaching and learning computer programming: Multiple research perspectives*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Mayer, R.E. (2002). *The promise of educational psychology. Volume II. Teaching for meaningful learning*. Upper Saddle River, NJ: Pearson Education, Inc.

McDonald, J.K., Yanchar, S.C., & Osguthorpe, R.T. (2005). Learning from programmed instruction: Examining implications for modern instructional technology. *Educational Technology Research & Development*, 53(2), 84-98.

Ninness, C., Rumph, R., McCuller, G., Harrison, C., Ford, A.M., & Ninness, S.K. (2005). A functional analytic approach to computer-interactive mathematics. *Journal of Applied Behavior Analysis*, 38, 1-22.

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

Roberts, E. (2004). Resources to support the use of Java in introductory computer science. In *Proceedings*

of the 35th SIGCSE Technical Symposium on Computer Science Education, Norfolk, VA (pp. 233-234). Retrieved September 27, 2007, from <http://portal.acm.org/citation.cfm?id=971384>

Scriven, M. (1969). The case for and use of programmed text. In A.D. Calvin (Ed.), *Programmed instruction: Bold new adventure* (pp. 3-36). Bloomington, IN: Indiana University Press.

Shimizu, H., & McDonough, C.S. (2006). Programmed instruction to teach pointing with a computer mouse in preschoolers with developmental disabilities. *Research in Developmental Disabilities, 27*, 175-189,

Sidman, M. (2006). Fred S. Keller, a generalized conditioned reinforcer. *The Behavior Analyst, 29*(2), 235-242.

SIGITE. (2005). *Final report of the ACM SIGITE curriculum committee 2005 project*. Retrieved September 27, 2007, from [http://www.acm.org/education/curric\\_vols/IT\\_October\\_2005.pdf](http://www.acm.org/education/curric_vols/IT_October_2005.pdf)

Skinner, B.F. (1954). The science of learning and the art of teaching. *Harvard Educational Review, 24*, 86-97.

Skinner, B.F. (1957). *Verbal behavior*. New York: Appleton-Century-Crofts, Inc.

Skinner, B.F. (1958). Teaching machines. *Science, 128*, 969-977.

Truong, N., Bancroft, P., & Roe, P. (2005, June 27-29). Learning to program through the Web. In *Proceedings of the 10<sup>th</sup> Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, (IT-iCSE'05), Monte de Caparica, Portugal. ACM Press.

Tsang, A.C.W., & Chan, N. (2004). An online problem-based model for the learning of Java. *Journal of Electronic Commerce in Organizations, 2*(2), 55-64.

Tudor, R.M., & Bostow, D.E. (1991). Computer-programmed instruction: The relation of required interaction to practical application. *Journal of Applied Behavior Analysis, 24*, 361-368.

Vargas, E.A., & Vargas, J.S. (1991). Programmed instruction: What it is and how to do it. *Journal of Behavioral Education, 1*, 235-251.

Williams, L.A., Wiebe, E., Yang, K., Ferzli, M., & Miller, C. (2002). In support of pair programming in

the introductory computer science course. *Computer Science Education, 12*(3), 197-212.

Xu, D., Wang, H., & Wang, M. (2005). A conceptual model of personalized virtual learning environments. *Expert Systems with Applications, 29*, 525-534.

## KEY TERMS

**Behavior Analysis:** A science and technology of behavior that seeks to understand the causes of behavior in terms of specifiable and directly measurable antecedents that account for and that determine performance at the level of the individual organism.

**Interteaching:** A dyadic interaction in which two learners come prepared to assess each other's understanding of a unit of knowledge and to teach each other, as needed, to a mutually informed level of competency.

**Learn Unit:** A contingency of reinforcement that includes an occasion for learning, a requirement for a learner to respond, and a consequence that confirms response accuracy or that provides remedial action until accuracy occurs.

**Personalized System of Instruction:** A comprehensive learning environment that permits the individual student to move through a progression of steps to competency at his or her own pace. The written word, rather than a lecture, is emphasized as the medium to transmit information to students. A student proctor "expert" verifies, within the context of an interpersonal interaction, a learning student's satisfactory completion of a step or recommends that studying continue until mastery of a step is demonstrated.

**Programmed Instruction:** A method for organizing knowledge for learning in incremental steps or frames of information where progress across successive steps requires demonstrated mastery at the level of the single step or frame.

## ENDNOTES

<sup>1</sup> Information about Java technology, including the Java programming language, is available at Sun Microsystems, Inc. (<http://java.sun.com/>).

<sup>2</sup> Final project example: <http://userpages.umbc.edu/~emurian/learnJava/swing/example/Main-Program.html>

<sup>3</sup> <http://www.behavior.org/index.cfm>

<sup>4</sup> This book is available online at the Cambridge Center for Behavioral Studies (<http://www.behavior.org/education/index.cfm>)

<sup>5</sup> The tutoring system, together with the source code and all instructional and assessment material, is freely available on the Web at <http://nasal.ifsm.umbc.edu/learnJava/tutorLinks/TutorLinks.html>