Behavior Analysis and ICT Education: Teaching Java™ with Programmed Instruction and Interteaching

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[1] 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.[2]

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[3] 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