Programmed Instruction and Interteaching Applications to Information Technology Education

Henry H. Emurian
Information Systems Department
College of Engineering and Information Technology

UMBC
1000 Hilltop Circle
Baltimore, Maryland 21250
Programmed Instruction and Interteaching Applications to Information Technology Education

Henry H. Emurian
UMBC
&
Ashley G. Durham
Centers for Medicare and Medicaid Services
&
Miji Mathews, Jingli Wang,
Amy Hu, Valeri Scott, Peng Zhang
John Goodall, Xin Li, Diana Wang, & Lidan Ha
UMBC
&
Heather Holden & Amy Abarbanel
UMBC
Programmed Instruction and Interteaching Applications to Information Technology Education: From Novice to Journeyman… and Beyond
For President-Elect

Richard W. Malott
Professor of Psychology, Western Michigan University
Ph.D. in Experimental Psychology, Columbia University

Recent Publications

Other Professional Positions and Activities
Award for Public Service in Behavior Analysis (2002).

Founder and Co-Chair, Teaching Behavior Analysis Special Interest Group of the ABAI (1993 - 1997).
Chair, Education Board of the ABAI (1993 - 1997).
Chair, ABAI Program Committee (1978 - 1980).
Secretary-Treasurer for the ABA (1974 - 1978).
One of the four co-founders of ABAI (now known as MABA, previously known as MABAI) (1974).
And, most important, originator of the Performing Arts (aka the Behavioral Bash).

Statement of Goals
Like everyone who’s ever run for president of ABAI, I’d encourage our continued, active support of the experimental and applied analysis of behavior and the practice of behavior analysis. I’d actively support consumer involvement in ABAI and our heavy emphasis on autism research and practice, seeing this heavy emphasis as supporting, not threatening BAC. I’d actively support ABAI’s Practice Board and its working with the Association of Professional Behavior Analysts and the Behavior Analysis Certification Board, seeing the BCBA as a major contributor to the impressive growth of ABAI’s attendance and the impressive growth in the number of behavior analysis M.A. programs throughout the USA and the rest of the world. I’d actively support ABAI’s efforts at the internationalization of behavior analysis, seeing that these efforts do not detract either financially or organizationally from our efforts of promoting the practice and the science of behavior analysis in the USA. I’d try to make attendance at the Behavioral Bash mandatory and forbid ABAI’s speakers’ from boring their audiences by reading their presentations, especially word-for-word from Power Points. And, I’d require/encourage all superstar invited speakers to spend 2 hours at a poster session admiring student posters and giving feedback to their creators—really.
Letters to the Editor

To the Editor:
Re "A Lost Eloquence," by Carol Muske-Dukes (Op-Ed, Dec. 29):

The balking of students to putting verse to heart by rote memorization is not limited to poetry. There is almost a pedagogical malaise that decries rote learning in disciplines like science, mathematics and engineering. And critical analysis and scholarship are being replaced by searching the Web.

There is a growing contempt for the hard work of achieving mastery.

But the beauty of a poem, once learned, is not in the recitation of words. The poem, committed to memory, becomes a vehicle of communion for the self and the soul. Rote learning of the tools of thought has similar benefits in all fields.

HENRY H. EMURIAN
Baltimore, Dec. 29, 2002

To the Editor:
As a 22-year-old recent college graduate, I applaud "A Lost Eloquence," by Carol Muske-Dukes (Op-Ed, Dec. 29), about the lost tradition of learning poems through memorization.

In this day and age, I was lucky enough to have a high school French teacher who demanded that we memorize and recite French poetry and fables. As students, we were given extra points for dramatic effusion.

Although, I am sad to say, my French skills are no longer stellar, the poems of Jacques Prevert and others still live in my blood.

I only wish that my English teachers had done the same.

KATE FILMORE
Brooklyn, Dec. 29, 2002
The New York Times

May 1, 2009
Op-Ed Columnist
Genius: The Modern View
By DAVID BROOKS

The latest research suggests a more prosaic, democratic, even puritanical view of the world. The key factor separating geniuses from the merely accomplished is not a divine spark. It’s not I.Q., a generally bad predictor of success, even in realms like chess. Instead, it’s deliberate practice. Top performers spend more hours (many more hours) rigorously practicing their craft.

We construct ourselves through behavior. As Coyle observes, it’s not who you are, it’s what you do.

http://www.nytimes.com/2009/05/01/opinion/01brooks.html
I am right.
The organism is always right.
The student is always right.
The organism is always right.
The student is always right.
Verbal Behavior

1957

The Keller Plan Handbook

1974

Essays on a Personalized System of Instruction

Fred S. Keller

J. Gilmour Sherman
Making a Science of Education

Bruce Alberts is the Editor-in-Chief of Science.

For decades, our prognosis has been grim—our complex, competing, and dangerous world demands that we must strive to be a science of education. Our education and social systems must be structured to select those with the most talent, energy, wisdom, and character as the next generation of leaders for our new segment of society. When I was young, I was taught that providing equal opportunities for everyone was a matter of social justice—part of the social contract in the United States. Now, I believe that it is also a matter of national survival. Any country that fails to encourage and develop the talent in each individual through a public school system will suffer greatly, because the quality of a nation depends on the collective wisdom of its leaders and its citizens.

An outstanding education system imparts values that support good citizenship, while empowering adults to be lifelong learners and problem solvers who can make wise decisions for their families, for their communities, and for their workplaces. Such an education system must continually evolve to remain relevant to the interests and needs of each new generation. To achieve these ambitious goals, we will need a common emphasis on both science education and the "science of education." It is my hope that science can help to promote progress on both fronts.

In 2008, Science began a monthly Education Forum. We hope to build on this strong beginning by recruiting high-quality articles on education from the worlds best experts for every section of the magazine. Thus we will be publishing important work in education, as Perspectives, Policy Forums, Reviews, or as original Research Reports and Articles, while continuing to cover education in the News section. This first issue of 2009, with its focus on Education and Technology (see page 53), represents a start that will hopefully inspire many more articles in the future.

As this special issue explains, the computer and communication technologies that have profoundly altered many other aspects of our lives seem to hold great promise for improving education as well. But technology is only a tool. To fully realize its promise for education will require a great deal of high-quality research, focused on its application and effects in both schools and non-school settings. Only by collecting and analyzing data on student learning can we hope to sort out the many variables that determine effectiveness.

The same type of scientific research is also needed to explore, analyze, and improve each of the many other components of educational systems. For example, the most important element of any educational system is a highly skilled teacher. Teacher recruitment, preparation, retention, and professional development all need to be informed by scientific research in education. Curriculum, pedagogy, assessment, and school systems management all require rigorous research. We hope that what is written in these pages about each of these important aspects of education will be reported and reviewed in Science.

Research in the social sciences is especially challenging because of the conditionalism of its findings. The history of social science is marked by its failure to follow through on many predictable theories that have been studied and analyzed. Some students may therefore question whether the science of education deserves a prominent place in this prestigious journal. For them, I offer the wisdom of Alfred North Whitehead, who wrote 80 years ago: "The art of education is not easy. It means in difficulties, especially those of elementary education, is a task worthy of the highest respect." (This is the reason why I am willing to work on it, and I wish to make it easier to work on it with others.)
• Among other things, I teach Java to Information Systems (IS) majors.
```java
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;
public class MyProgram extends JApplet {
    JLabel myLabel;
    public void init() {
        myLabel=new JLabel("This is my first program.");
        getContentPane().setBackground(Color.YELLOW);
        getContentPane().add(myLabel);
    }
}
```
This is my first program.
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. }

• Near transfer (recite & “understand”)
• Far transfer (meaningful learning → solve novel problems)
  ➢ 12 “rules” questions
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;

public class MyProgram extends JApplet{
  JLabel myLabel;
  public void init()
  {
    myLabel = new JLabel("This is my first program.");
    getContentPane().setBackground(Color.YELLOW);
    getContentPane().add(myLabel);
  }
}

Intraverbal performances
“Ordered Tuple”
Chain
```
import javax.swing.JApplet, javax.swing.JLabel;
import java.awt.Color;

public class MyProgram {
    public void init() {
        JLabel myLabel = new JLabel("This is my first program.");
        getContentPane().add(myLabel);
        getContentPane().setBackground(Color.YELLOW);
    }

    public static void main(String[] args) {
        javax.swing.JApplet applet = new javax.swing.JApplet();
        applet.init();
    }
}
```

"Semantic networks"
Hierarchical Relational Frames
Combinatorial Entailment

Constructor method

add(myLabel)
getContentPanel()
init()
setBackground(Color.YELLOW)

class

MyProgram
JLabel
JApplet
Color

separator

.
;
{
}

keyword

class
class
import
import
new
new
public
public
void
void
extends
extends

assignment operator

=

object

myLabel

YELLOW
Learn Unit: Greer & McDonough (1999)

Gagne’s Hierarchical Model.

Learn Program

Learn Items

- Identify Item
- Type Item

Learn Rows

- Identify Rows
- Type Rows

Meaningful Learning
Learn about the Alice interface and how to start creating your own worlds.

(This is an older video, and the intro states that Alice is only available for PC. This is no longer the case, as Alice is available for PC, Mac, and Linux.)

http://www.alice.org/
Without Alice, at risk CS1 students average a **C grade**
...and only **47 percent** go on to take CS2.

With Alice, at risk CS1 students average a **B grade**
...and **88 percent** go on to take CS2.

M. Maskar, D. Ume, and S. Cooper
“Evaluating the Effectiveness of a New Instructional Approach”
Tutorials for the First-Time Computer User

JANAN AL-AWAR, ALPHONSE CHAPANIS, AND W. RANDOLPH FORD

Abstract—This paper describes a general methodology and principles for the preparation of tutorials, or computer-assisted instructional courses, to introduce first-time users to computer terminals. The methodology and principles are especially designed to prepare tutorials that will make computers seem friendly and that will motivate casual or discretionary users to learn more about computers. Examples are drawn from a tutorial prepared for the IBM 3277 Display Station.

users that they can communicate with a computer easily and effectively. Our tutorial uses the computer as an adaptive teaching system. It is adaptive in the sense that it (a) allows students, or users, to proceed at their own pace, and (b) introduces variations in the presentation of materials according to the student’s performance. Variations are made through branches that are controlled by the user's responses.

Fig. 1. Keyboard of the IBM 3277 Display Station, Model 2.

Fig. 2. Operator at the terminal used in testing the tutorial program.
Why won’t they (students) respond?

- In comparison to Computer Science (CS) students, Information Systems (IS) students exhibit a low rate of computer programming.
• Students in Information Systems (IS) do not like to write computer programs.
• IS students have **minimal coursework** in computer programming and programming languages.
• IS students **need** a fundamental mastery of programming principles, especially related to the object-oriented paradigm.
• IS students are often **demoralized** by taking courses with computer science majors taught by computer science faculty.
• How can we best **help** IS students achieve the objective?
Design-Based Research Methods for Studying Learning in Context: Introduction

William A. Sandoval  
Graduate School of Education and Information Studies  
University of California, Los Angeles

Philip Bell  
Cognitive Studies in Education  
University of Washington

The field of psychology has a long history of interaction with education, and educational psychology has had a profound impact on how issues of learning have been framed and studied in educational contexts. Still, it has never been simple to translate theoretical insights into educational practice. Educational psychology has been criticized for not creating “usable knowledge” (Biggeman, 2002). Currently, an educational psychology that is both usable in a practical sense and scientifically trustworthy cannot proceed without directly studying the phenomena it hopes to explain in its inherent messiness. A little over a decade ago, Brown (1992) described her evolving approach to “design experimentation” as an effort to bridge laboratory studies of learning with studies of complex instructional interventions based on such.

http://www.designbasedresearch.org/
Extinction?
• Observe students in context
  – Repeated observations in one classroom with one group of students and one instructor
• Improve the instructional design over successive replications
  – Systematic replication (Sidman, 1960)
• Emphasizes movement of all students to a common learning outcome (*True Gain*)
  – Contrasts with between-group studies concerned with effect size differences

• *It is more meaningful to hold constant the level of mastery required and look at differences in time to achieve that level. This reflects the true gain of an educational technique* (p. 185).
What state is steady? (When is the gain true?)

Power Function

Errors

Practice Trials
• **Analytic behavioral application is the process of applying sometimes tentative principles of behavior to the improvement of specific behaviors** (Baer, Wolf, & Risley, 1968, p. 91).


• [http://seab.envmed.rochester.edu/jaba/articles/1968/jaba-01-01-0091.pdf](http://seab.envmed.rochester.edu/jaba/articles/1968/jaba-01-01-0091.pdf)
“Competing Responses”
Randomized Field Trial

http://ies.ed.gov/ncee/wwc/
Counterpoint

http://www.nifdi.org/
Report of the What Works Clearinghouse Expert Panel

To: National Board for Education Sciences

From: Hendricks Brown, Ph.D.
    David Card, Ph.D. (chair)
    Kay Dickerson, Ph.D.
    Joel Greenhouse, Ph.D.
    Jeffrey Kling, Ph.D.
    Julia Littell, Ph.D.

Re: Expert Report on the What Works Clearinghouse

Date: October 21, 2008

1. Introduction and Summary

We have been charged with the task of conducting a "... focused study addressing the fundamental question of whether the Clearinghouse's evidence review process and reports are scientifically valid-that is, provide accurate information about the strength of evidence of meaningful effects on important educational outcomes." (Our complete charge is reproduced as Appendix A, below).

Based on our investigation and analysis of the What Works Clearinghouse (hereafter, WWC), we have concluded that:

(1) WWC procedures and processes for identifying and extracting information from intervention studies are generally well documented and follow reasonable standards and practices for systematic reviews;

(2) WWC Intervention and Topic Area Reports provide succinct and meaningful summaries of the evidence on the effectiveness of specific education interventions.

Support for these conclusions is detailed in the remainder of the report. We have also formed a number of specific recommendations for the continued enhancement and improvement of WWC procedures, which are summarized in section IV. Primary among these recommendations is that the Department of Education commission a comprehensive review of the full range of WWC activities and procedures, with a time frame to allow a complete consideration of a number of issues we have not been able to fully evaluate in this report.

Keep Making Responses

- Principles to promote retention and transfer:
  - Repeated practice with different instructional modalities (Halpern & Hakel, 2003)
  - Socially supported interactions (Fox & Hackerman, 2003)
1. A set of **structured interactions** between a learner and a tutor.

2. Occasions **disciplined study behavior** that is focused on the individual learner.

3. Manages the **moment-by-moment interactions** between a learner and a tutor: *learn units*.

4. A **step-wise progression** from elementary facts to the achievement of meaningful learning.

---

**Java Tutor**

[http://nasa1.ifsm.umbc.edu/learnJava/tutorLinks/SwingTutorLinksV2.html](http://nasa1.ifsm.umbc.edu/learnJava/tutorLinks/SwingTutorLinksV2.html)
Ten Features of PI
(Holland, 1960; Scriven, 1969; Skinner, 1958; Vargas & Vargas, 1991)

1. Comprehensibility of each unit or “frame,”
2. Tested effectiveness of a set of frames,
3. Skip-proof frames,
4. Self-correcting tests,
5. Automatic encouragement for learning,
Ten Features of PI
(Holland, 1960; Scriven, 1969; Skinner, 1958; Vargas & Vargas, 1991)

6. Diagnosis of misunderstandings,
7. Adaptations to errors by hints, prompts, and suggestions,
8. Learner constructed responses based on recall,
9. Immediate feedback, successive approximations to a terminal objective, and
10. Student-paced progress.
Teaching Machines

From the experimental study of learning devices which arrange optimal conditions for self-instruction.

B. F. Skinner

There are more people in the world than ever before, and for a greater part of them, there is an education. The demand cannot be met simply by building more schools and training more teachers. Education must become more efficient. To this end curricula must be revised and simplified, and methods and techniques improved. In any other field a demand for increased production would have led to an invention of better and less expensive machines. In education, however, the so-called audio-visual aids are being mass-produced: film projectors, television sets, photography, and tape recorders are finding their way into American schools and colleges.

Audio-visual aids are devices that can present visual and sometimes auditory material to the student and, when successful, make it clear and interesting to the student. There are other functions which they can serve: they can act as a substitute for lecture or writing. It has been seen in the productive interchange between teacher and student in the small classroom, that interaction is a function which the instructor to a large extent must perform in order to teach large numbers of students. There is a real danger that it will be wholly absent if use of equipment is designed simply to present material becomes widespread. The teacher is becoming more and more a passive receiver of information.

Pressey's Teaching Machines

There is another kind of teaching device which will discipline the student to take an active role in the instructional process. The possibility was recognized in the 1920s, when S. L. Pressey designed several machines for the automatic testing of intelligence and information. A recent model of one of these is shown in Fig. 1. In using the device, the student places his answer to a numbered item in a multiple-choice test. He proceeds to the next item if he is wrong, the cover is opened, and he must continue to make choices until he is right (1). Such machines, Pressey pointed out (2), could not be set to score, nor could they grade. When an examination is completed and scored, the information is returned to the student. The student's performance is the basis for future instruction. The instructional system is supported by a self-teaching device, however, can have an important instructional effect. Pressey also pointed out that such machines would increase efficiency in another way. Even in a small classroom the teacher usually knows that he is not teaching in an environment in which his information is passed to his students. Those who could go faster are penalized, and those who should go slower are penalized and unnecessarily punished by criticism and failure. Machine instruction would permit each student to proceed at his own pace.

The "industrial revolution in education" which Pressey envisioned modestly refers to the idea of teaching in a machine. In 1958 he expressed his disappointment (3):

"The problem of invention is relatively simple," he wrote. "With a little money and engineering resource, a great deal could really be done. The writer has found from bitter experience that one person alone can accomplish relatively little and he is especially dropping further work on these problems. But he hopes that enough may have been done to stimulate other workers that this fascinating field may be developed.

Pressey's machines accumulated to date in normal classes: the world of education, was not ready for them. But that they had limitations which probably contributed to their failure. Pressey was working against a background of psychological theory which had not even begun with the learning process. The study of human learning was dominated by the "memory drum" and similar devices originally designed to study forgetting. Rate of learning was observed, but little was done to change it. With the subject of such an experiment a failure to learn at all was of little interest. "Practical" and "impractical" theories of learning, and principles of "measured and spaced practice," concerned the condition under which processes were remembered. Pressey's machines were designed against this theoretical background. As long as the memory drum, they were primarily testing devices. They were to be used after some amount of learning had already taken place elsewhere. By confirming correct responses and by reinforcing correct responses which should not have been acquired, a self-teaching machine does, indeed, teach, but it is not designed primarily for that purpose.

Necessary, Pressey seems to have been the first to emphasize the importance of immediate feedback in instruction and to propose systems in which each student should go slower and more swiftly, and who.

EDUCATIONAL RESEARCH AND STATISTICS

A MACHINE FOR AUTOMATIC TEACHING AND DRILL MATERIAL

In a previous number of this journal, the writer described a "simple apparatus which...

S. L. Pressey

1926

1958

Dr. Skinner is Education Professor of psychology at Harvard University, Cambridge, Mass.

OCTOBER 1958
As public purse strings tighten, the day may come when learning time and learning costs are subjected to close accountability in public schools and university education also.

The Programmed Instruction Era: When Effectiveness Mattered

By Michael Molenda

Programmed instruction (PI) was devised to make the teaching-learning process more humane by making it more effective and expeditious to individual differences. B.F. Skinner's original prescription, although it met with some success, had serious limitations. Later innovators improved upon the original notion by incorporating more human interaction, social reinforcers, and other forces of feedback, larger and more flexible chunks of instruction, and more orientation to learner appeal. Although PI itself has receded from the spotlight, technologies derived from PI such as programmed tutoring, Direct Instruction, and Personalized System of Instruction have compiled an impressive track record of success when compared to so-called conventional instruction. They paved the way for computer-based instruction and distance learning. The success of the PI movement can be attributed largely to the commitment of its proponents to relentless, objective measurement of effectiveness.

Origins of the Programmed Instruction Movement

During the first half of the 20th century, research and theory in American psychology tended to revolve around the perspective of behaviorism, and Thorndike (1911) theorized—

the law of using the law of effect, and the law of exercise—remained at the center of discussion for decades. In the 1920s Sidney Pressey, a psychology professor at Ohio State University, invented a mechanical device based on a typewriter drum, designed primarily to automate testing of simple intellectual material (1926). As he experimented with the device he realized that it could also provide control over drill-and-practice exercises, teaching as well as testing. Explanations for why his device was successful are explicitly drawn upon Thorndike's laws of re- doing, effect, and exercise as theoretical rationales (Pressey, 1937). Unfortunately, despite the fact that Pressey continued to develop successful self-teaching devices, including punchboards, that had all the qualities of later 'teaching machines,' his efforts were essentially a dead end in terms of lasting effect on education. However, Pressey lived and worked long enough to participate in the discussions surrounding the new generation of teaching machines that came along in the 1950s.

The innovations that had a more enduring impact on education and training was animated by a rethinking of Thorndike's behavioristic principles under the label of radical behaviorism. This school of thought proposed a more rigorous definition of the law of effect, adopting the term consequence to refer to any event that increases the frequency of a preceding behavior. Operant conditioning, the major operationalization of this theory, involved the relationships among stimuli, responses, and the consequences that follow a response (Skinner, 1938). The leading proponent of radical behaviorism, B.F. Skinner, demonstrated that by manipulating these three variables experimenters
Teaching Machines

From the experimental study of learning-computer devices which arrange optimal conditions for self-instruction.

B. F. Skinner

There are more people in the world today who believe what they feel than there have ever been before. The sphere of what they feel is being rapidly expanded, and the limits of human experience and knowledge are being transcended. The ideal of a world of superior learning conditions becomes more realizable as the boundaries of human achievement are extended.

Skinner's Teaching Machines

There is another kind of educational apparatus which will encourage these tendencies to take an active role in the instructional process. This possibility was recognized in 1950, when B. F. Skinner described several machines by the name of "teaching devices" which might serve as teaching tools in the process of learning. This article describes some of these machines.

"The original AutoTutor, released in the early 1960s, provided individualized instruction long before general-purpose desktop computers were feasible."

50+ Years

By Michael Meehle

Programmed instruction (PI) was devised to make the teaching-learning process more efficient by making it a more effective and systematic instruction of individuals. PI is essentially a programmed text, designed primarily to improve the teaching-learning process. It is a form of self-instruction, in which the learner works through a sequence of lessons, with each lesson designed to help the learner master the material covered in the previous lesson.

The effectiveness of PI was measured by comparing the performance of students taught with PI to the performance of students taught by traditional means. The results showed that students taught with PI performed better than those taught by traditional means.

The Programmable Instruction Movement

During the first half of the 20th century, many educators and psychologists believed that the use of mechanical devices could improve the teaching-learning process. These devices were called "teaching machines," and they were designed to help students learn in a more efficient and effective manner.

The Programmable Instruction Movement was characterized by the use of mechanical devices, such as the M-T-1, AutoTutor, and others, to help students learn. These devices were designed to provide individualized instruction, in which the learner works at their own pace and in their own way.

The Programmable Instruction Movement was a significant step forward in the development of educational technology, and it paved the way for the development of modern educational technology, such as computer-based instruction and distance learning.
The immediacy of reinforcement did not prove to be critical for a great many types of learning tasks. Indeed, the efficacy of “knowledge of results” as a reinforcer did not stand up under scrutiny. In retrospect, it was predictable that ‘knowledge of correct response’ would not work as a universal reinforcer. Researchers (and lay people) already knew that different people respond to different reinforcers at different times. When a person is satiated with ice cream, ice cream is no longer reinforcing. The same is true of being told the correct answer. At some point curiosity is satiated.
The immediacy of reinforcement did not prove to be critical for a great many types of learning tasks. Indeed, the efficacy of “knowledge of results” as a reinforcer did not stand up under scrutiny. In retrospect, it was predictable that ‘knowledge of correct response’ would not work as a universal reinforcer. Researchers (and lay people) already knew that different people respond to different reinforcers at different times. When a person is satiated with ice cream, ice cream is no longer reinforcing. The same is true of being told the correct answer. At some point curiosity is satiated. Researchers rediscovered that there are no universal reinforcers (p. 55).

The Programmed Instruction Era: When Effectiveness Mattered

By Michael Molenda

Programmed instruction (PI) was designed to make the teaching-learning process more effective by making it more efficient and systematic. E.L. Thorndike’s original conception, although met with some success, had serious limitations. Later innovations improved upon the original notion by incorporating a range of human interactions, social reinforcers and other forms of feedback, larger and more extended cycles of instruction, and more attention to learner appeal. Although in itself it has evolved from the spotlights, technologies derived from it, such as programmed instruction, make instruction and remediated systems of instruction have compiled several impressive records of success when compared to older conventional instruction. They paved the way for computer-based instruction and distance learning. The success of the PI movement can be attributed largely to the commitment of its propagators to achieving objective measurement of effectiveness.

Origins of the Programmed Instruction Movement

During the first half of the 20th century, research and theory in American psychology tended to revolve around the principles of behaviorism, and Thorndike’s (1911) stimulus–response theory—the law of effect, the law of effect, law of effect, and the law of effect—continued to be the center of discussion for decades. In the 1950s, Baer and a psychology professor at Ohio State University, invented a mechanical device based on a typewriter drum, designed primarily to assist students in remembering simple division material (1956). As he experimented with the device he realized that it could also provide control over drills and practice exercises, teaching as well as testing. As explained, why he wanted to test the device was successful, he explicitly drew upon Thorndike’s law of effect, and similar to the theoretical principles (Baer, 1957). Unfortunately, despite the fact that Thorndike continued to develop successful self-teaching devices including pinboards, that had all the qualities of later “teaching machines,” some of his efforts were eventually a dead end in terms of a lasting effect on education. However, many of his ideas lived and were long enough to influence the decisions concerning the new generation of teaching machines that came along in the 1960s.

The movement that had its roots during the peak of instruction and teaching was initiated by a refocusing of the relationship between principles and the label of educational behaviorism. This kind of relationship was more rigorous and contained the law of effect, adopting the term reinforcement to describe an event that increases the frequency of a preceding behavior. Operator is an individual who understands the concept of reinforcement, and Thorndike’s (1911) teacher—
Programmed Instruction Tutoring System
Introduction

You will experience the benefits of guided practice in the construction and execution of a Java applet. In this tutor, the program will be based on the JApplet class. An applet is a Java computer program that is downloaded from a server computer and run by your browser. In this tutor, the program will display the string "This is my first program." in the browser window. You will learn to write and to understand the Java code to produce the program.

The program that you will learn will be organized within the following groups of statements:

1. import system files to use in your program.
2. define your program as a subclass of the JApplet class.
3. declare a JLabel object to display the string.
4. insert a method, named init(), in your program to execute the code, and
5. create and display the object.

To see the JApplet running, select the Start the JApplet button displayed below, and be prepared to wait a few moments until a new

• Advance organizers
  – Template of a Java Applet
• Observe the applet in action
The lines displayed in the adjacent box consist of lines of Java code. This tutor will teach you to understand and to write the code in the program. You do not need to study the program that is displayed. The program is displayed for you now only to show you what you will be able to do when you complete the tutor.

Examine the adjacent lines of code to see the general appearance of a Java program and the types of symbols and expressions that appear. You are not expected to understand these lines of code yet.

The white space in a line is ignored by the compiler. The indentation, then, is to assist the visualization of the various statements, declarations, and methods that determine the composition of a Java program. The tutoring system will enforce some visualization.

```java
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;
public class MyProgram extends JApplet {
    JLabel myLabel;
    public void init() {
        myLabel = new JLabel("This is my first program.");
        getContentPane().setBackground(Color.YELLOW);
        getContentPane().add(myLabel);
    }
}
```
The lines displayed in the adjacent box consist of HTML tags and parameters to run the MyProgram.class program, which is produced by compiling the Java code. The lines are created with a text editor and saved as MyProgram.html. There is no compilation with the HTML file. It is used as it was written in the editor.

The Java class file, which is executed as a JApplet, is started by using MyProgram.html as the target file in the browser URL.

Examine the adjacent lines to familiarize yourself with the general appearance of an HTML file. You are not expected to understand these lines yet.

When you are ready to continue, select the Proceed button.
Find this: javax.swing.JApplet
Java Tutor: Item Learning

import java.awt.Button;
import java.awt.Label;
public class MyProgram extends JApplet {
    JLabel myLabel;
    public void init() {
        myLabel = new JLabel("This is my first program.");
        myLabel.setVisible(true);
        getContentPane().add(myLabel);
    }
}

You do not need instructions to use this tutor because the events are determined by the enabled buttons and by the accuracy of your typed input and other selections that you make. If you cannot create the Java item when asked to type it, simply press the Enter key with the cursor in the text area.
The first line of code is this:

`import javax.swing.JApplet;`

The Java item being taught is highlighted in blue.

`import javax.swing.JApplet;`

The Java programs that you write will almost always be based on pre-written Java programs, known as class files. Since these pre-written, or built-in, Java files are not in your current directory, you must insert the Java code in your program that informs the compiler about the location of those built-in Java class files on the system that you are using. The `import` term is used to do that. In the Java Language Specification, the character sequence `import` is classified as a keyword, which means that it is reserved for use.

The `import` keyword will be followed by a directory path and the name of a built-in Java class file. By using the `import` keyword first in your program, you can later refer to a built-in Java class file with a shorthand notation. This means that you will not have to write the entire directory path each time you use a built-in Java class file in the Java programs that you write.

Which one of the following statements best describes the `import` keyword?

- Write a shorthand notation for the compiler to locate a built-in Java class file on the system.
- Allow a modified reference to the `import` file in your current directory.
- Save a file in the `import` directory.
- Prepare to delete all instances of an imported object.

Correct Choice: Write a shorthand notation for the compiler to locate a built-in Java class file on the system.
• Davis, Bostow, and Heimisson (2007) reported the inclusion of abstract statements of a behavioral relation (a “rule”) in many frames of a programmed instruction tutor designed to promote generalization of what was taught in the tutor.

The expression `javax.swing.JLabel` refers to a file named `JLabel.class`. This is a file that is used to construct instances of the `JLabel` class. Since your program will construct an instance of the `JLabel` class, it is easier to use a shorthand notation to refer to the `JLabel.class` file.

When you write

```java
import javax.swing.JLabel;
```

at the beginning of the program, this allows the later use of the built-in `JLabel` class file by the shorthand notation, which is `JLabel`, by itself.

The `JLabel` class file, `JLabel.class` as it exists in a directory, contains the compiled Java code to display text on the screen, and it is located in the `javax.swing` package on the system. You can think about a package as a directory in which related files are stored. The reason that you have to use either `import` or the full path, `javax.swing.JLabel`, is simply because the `JLabel.class` file is not in your directory. That file is in a different directory on the system, and the compiler needs to know where it is located before it can be used in your program.

Notice that `JLabel` begins with a capital letter. That tells you that it is a class file in Java. **That is an important rule to know.** The definition of a class is presented later.
• An ongoing challenge relates to the optimal design of the content of frames – the “explanations” (Wittwer & Renkl, 2008). A challenge for programmed instruction is to develop frames of information that are effective for learning, and that requires a conceptual framework for understanding the effectiveness of instructional explanations.

Your choice was correct.

The import term (or import item) is a Java keyword that is used to allow the programmer to write Java code later to access a class that is not in your current directory without having to type the entire path for the compiler to find that class file on the system.

Type the Java here, and press Enter.
method
  - add(myLabel)
  - getContentPane()
  - init()
  - setBackground(Color.YELLOW)

Constructor method
  - JLabel("This is my first Program."")

class
  - MyProgram
  - JLabel
  - JApplet
  - Color

separator

object
  - myLabel
  - YELLOW

keyword
  - class
  - import
  - new
  - public
  - void
  - extends

assignment operator
  - =
Line familiarity and identification.
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;
public class MyProgram extends JApplet {
    JLabel myLabel;
    public void init() {
        myLabel = new JLabel("This is my first program.");
    }
}
The seventh line of code is this:

```
myLabel = new JLabel("This is my first program.");
```

The code in line 7 is a constructor statement. This line constructs a new instance of the JLabel class, and the name of the new instance is myLabel. Once the instance is constructed, it exists in a special area in computer memory. The fact that an instance exists does not mean that it can be viewed. The instance must be installed within a container, such as a JApplet container, which has a content pane layer to hold objects. In the present program, the container is MyProgram, which is a subclass of the JApplet class.

Note that the instance myLabel is constructed as a new instance of the JLabel class. Inside the JLabel class is a method JLabel(String a) that accepts the string argument that will be displayed whenever myLabel is added to a container.
E-assessment by design: using multiple-choice tests to good effect

David Nicol
University of Strathclyde, UK

Over the last decade, larger student numbers, reduced resources and increasing use of new technologies have led to an increased use of multiple-choice question (MCQs) as a method of assessment in higher education courses. This paper identifies some limitations associated with MCQs from a pedagogic standpoint. It then provides an assessment framework and a set of feedback principles that, if implemented, would support the development of learner self-regulation. This different uses of MCQs are then mapped out in relation to this framework using case studies of assessment practice drawn from published research. This analysis shows the different ways in which MCQs can be used to support the development of learner self-regulation. The framework and principles are offered as a way of helping teachers design the use of MCQs in their courses and of evaluating their effectiveness in supporting the development of learner autonomy. A key message from this analysis is that the power of MCQs (to enhance learning) is not increased merely by better test construction. Power is also achieved by manipulating the context within which tests are used.
Styles of Learning and Thinking Matter in Instruction and Assessment

Robert J. Sternberg, 1 Elena L. Grigorenko, 2 and Li-fang Zhang 3

1 Yale University, 2 Yale University, and 3 University of Hong Kong

ABSTRACT—There are two styles of learning and thinking: ability-based and personality-based. The former are assessed by maximum-performance tests, and the latter are assessed by typical-performance tests. We argue that both kinds of styles matter for instruction and assessment in school. In particular, shapingKnown differences in ability-based knowledge and understanding of the material learned in school are identical may nevertheless manifest their achievements differently. One may do better on a multiple-choice test measuring memory of facts, the other may do better on an essay test that encourages creative use of the material that has been learned. This may be a result of skill-based differences between the two students taking the two different kinds of tests, preference-based differences for the two kinds of tests, or both.

An essential point of this article is that there are both ability-based and personality-based styles that matter for instruction and assessment. Taking these styles into account can improve instruction and assessment. Not taking these into account prevents students from capitalizing on strengths and compensating for or correcting weaknesses and thus is suboptimal.

In this article, we discuss styles as a basis for understanding individual differences in how people learn and think. First, we define what styles are. Then we describe how styles apply to two theories (Sternberg, 1997a, 1997b) and draw our conclusions.

DEFINING STYLES

We define styles here as individual differences in approaches to tasks that can make a difference in the way in which and potentially, in the efficacy with which a person perceives, learns, or thinks. We limit our definition of styles to those that matter for cognition because, in our view, that was the original intention of the “cognitive style movement”—identifying styles of processing information that are consequential for cognition (e.g., Cooper, 1979, 1985; Cooper, Besman, Duc-Allbert, & Fiore, 1964; Kviton, 1976; Kagan, 1965; Martin, 1960; Martin & Suchow, 1999; Witkin, Dyk, Faterson, Goodenough, & Karp, 1962).

The styles literature focuses on two specific aspects: differences referred to as “ability-based” and “personality-based” theories of styles (Sternberg, 1997b; Zhang & Sternberg, 2005, 2006). Styles and personality-based styles, although these terms do not totally capture the differences between them, Styles traditionally have been seen as being at the interface between cognition and personality (Sternberg, 1997a), and it probably stands to reason that their formulation and measurement have drawn on both the cognitive and personality literatures.

According to our definition, abilities and attributes measured by maximum-performance tests or by typical-performance tests

As this article, we use the term style somewhat differently from the way we have used it here previously. Here, we use it to refer either to a maximum-performance (ability-based) or a typical-performance (personality-based) difference or preference in learning or thinking that can lead to differential outcomes in instruction and assessment.

Address correspondence to Robert J. Sternberg, Yale School of Arts and Sciences, Batten Hall, New Haven, CT, University, New Haven, CT 06517; e-mail: Robert.Sternberg@yale.edu.

Copyright © 2008 Association for Psychological Science.
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;
public class MyProgram extends JApplet {
    JLabel myLabel;
}

Your choice was correct: thisLabel = new JTextField();

This is the only selection that does not violate any rule that you have learned. Even though the tutor did not teach the JTextField class, the constructor statement that you selected follows all the rules that you know for the identifiers, keywords and class names.
```java
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.Color;
public class MyProgram extends JApplet {
    JLabel myLabel;
    public void init() {
        myLabel = new JLabel("This is my first program.");
    }
}
```
Type the program in the white space below, and select the Submit button. Do not use the Tab key. Please do not use notes. Try to enter the program from your memory. If you can't remember the program, just select Submit, and you can see the code again.

```java
import java.awt.Color;
import java.awt.Label;
public class MyProgram extends JApplet {
    public void init() {
        JLabel myLabel = new JLabel("This is my first program.");
        getContentPane().setBackground(Color.YELLOW);
        getContentPane().add(myLabel);
    } 
```

Your input was not correct. Examine the below code and see if you can spot your error. The below code has the same format as the previous tutor code, but you do not have to use that format in the adjacent window. Select the Noted button when you are ready to try again.
1. **Lecture**
   - Repeat the tutor material while students write the code

2. Run the applet on the web
Interteaching

1. A **mutually probing, mutually informing conversation between two people** (Boyce & Hineline, 2002)

2. The questions on a topic to be addressed by the participants during a dialogue are prepared in advance by the teacher, and the **students come prepared to interteach**

3. Has the objective of insuring, by the participants as a team, that each **member of the dyad** can answer the questions with understanding
Interteaching: A Strategy for Enhancing the User-Friendliness of Behavioral Arrangements in the College Classroom

Thomas E. Boyce
University of Nevada, Reno

Philip N. Hineline
Temple University

“Interteaching” is an arrangement for college classroom instruction that departs from the standard lecture format and offers an answer to criticisms commonly directed at behavioral teaching techniques. This approach evolved from exploratory use of small-group arrangements and Ferster and Perrott’s (1968) “interview technique,” leading ultimately to a format that is organized around focused dyadic discussion. Specific suggestions are offered that might enable both seasoned and novice instructors to incorporate this or similar arrangements into their classrooms. This approach retains some key characteristics of Keller’s personalized system of instruction and precision teaching, but offers greater flexibility for strategies that are based on behavioral principles.

*Key words:* applied behavior analysis, education, instruction, interviewing, PSI, precision teaching, reciprocal peer tutoring
“GOOD-BYE, TEACHER . . .”

FRED S. KELLER

ARIZONA STATE UNIVERSITY

When I was a boy, and school “let out” for the summer, we used to celebrate our freedom from educational control by chanting:

Good-by scholars, good-by school; Good-by teacher, darned old fool!

We really didn’t think of our teacher as deficient in judgment, or as a clown or jester. We were simply escaping from restraint, dinner pail in one hand and shoes on the other, with all the delights of summer before us. At that moment, we might even have been well-disposed toward our teacher and might have felt a touch of compassion as we completed the rhyme.

“Teacher” was usually a woman, not always young and not always pretty. She was frequently demanding and sometimes sharp of tongue, ever ready to pounce when we got out of line. But, occasionally, if one did especially well in home-work or in recitation, he could detect a flicker of approval or affection that made the hour in class worthwhile. At such times, we loved our teacher and felt that school was fun.

It was not fun enough, however, to keep me there when I grew older. Then I turned to another kind of education, in which the reinforcements were sometimes just as scarce as in the schoolroom. I became a Western Union messenger boy and, between deliveries of telegrams, I learned Morse code by memorizing dots and dashes from a sheet of paper and listening to a relay on the wall. As I look back on those days, I conclude that I am the only living reinforcement theorist who ever learned Morse code in the absence of reinforcement.

It was a long, frustrating job. It taught me that drop-out learning could be just as difficult as in-school learning and it led me to wonder about easier possible ways of mastering a skill. Years later, after returning to school and finishing my formal education, I came back to this classical learning problem, with the aim of making International Morse code less painful for beginners than American Morse had been for me (Keller, 1948).

During World War II, with the aid of a number of students and colleagues, I tried to apply the principle of immediate reinforcement to the early training of Signal Corps personnel in the reception of Morse-code signals. At the same time, I had a chance to observe, at close hand and for many months, the operation of a military training center. I learned something from both experiences, but I should have learned more. I should have seen many things that I didn’t see at all, or saw very dimly.

I could have noted, for example, that instruction in such a center was highly individualized, in spite of large classes, sometimes permitting students to advance at their own speed throughout a course of study. I could have seen the clear specification of terminal skills for each course, together with the carefully graded steps leading to this end. I could have seen the demand for perfection at every level of training and for every student; the employment of classroom instructors who were little more than the successful graduates of earlier classes; the minimizing of the lecture as a teaching device and the maximizing of student participation. I could have seen, especially, an interesting division of labor in the educational process, wherein the non-commissioned, classroom teacher was restricted to duties of guiding, clarifying, demonstrating,
In Support of Pair Programming in the Introductory Computer Science Course

Laurie Williams¹, Eric Wiebe², Kai Yang¹, Miriam Ferzli², and Carol Miller¹
North Carolina State University, Raleigh, NC, USA, ¹Department of Computer Science, and ²Department of Math, Science and Technology Education

ABSTRACT

A formal pair programming experiment was run at North Carolina to empirically assess the educational efficacy of the technique in a CSI course. Results indicate that students who practice pair programming perform better on programming projects and are more likely to succeed by completing the class with a C or better. Student pairs are more self-sufficient which reduces their reliance on the teaching staff. Qualitatively, paired students demonstrate higher order thinking skills than students who work alone. These results are supportive of pair programming as a collaborative learning technique.

J. EDUCATIONAL COMPUTING RESEARCH, Vol. 17(1) 19-46, 1997

THE PSYCHO-SOCIAL PROCESSES AND COGNITIVE EFFECTS OF PEER-BASED COLLABORATIVE INTERACTIONS WITH COMPUTERS*

JIHN-CHANG J. JEHNG
National Central University

ABSTRACT

This research project consists of two related studies involving first- and second-year university students learning to write recursive programs. The first employed a micro-structure analysis that examined the psycho-social processes underlying peer-based interactions in two different computer-based collaborative learning environments: face-to-face vs. distributed context. These processes may be viewed as knowledge building activities that occur in three key collaborative situations: communication, negotiation, and consolidation. Results of this study demonstrated the two collaborative learning environments produced two distinct psycho-social behaviors manifested by the students. In the second study, 130 students were divided into four groups. Three participated in collaborative learning environments; the fourth made up a control group whose members learned in isolation from one another. All the students learned to write recursive programs for designing geometric patterns. Although results indicated the two groups of students did not show significant differences in their program evaluation and completion abilities, students who participated in three collaborative learning environments demonstrated superior program generation abilities on the posttest compared to those who had learned to solve problems individually.

An increasing number of educational and psychological studies are focusing on forms of peer-based interactions with computers [1-6]. Approaches for peer-based interactions with computers differ little from those for any other peer-based group.

*This research project was supported by a grant from the National Science Council, Taiwan, under the contract NSC83-01105-032.004. The author is grateful to Dr. Tak Wai Chan for his assistance with part of this research.
ARE TWO HEADS BETTER THAN ONE FOR SOFTWARE DEVELOPMENT? THE PRODUCTIVITY PARADOX OF PAIR PROGRAMMING

By: VenuGopalBalaji
College of Business
Prairie View A&M University
Prairie View, TX 77446
U.S.A.
veballaji@pvamu.edu

Radhakant Mahapatra
College of Business Administration
University of Texas at Arlington
Arlington, TX 76019-0437
U.S.A.
mahapatra@uta.edu

Srithar Narasimhan
College of Business Administration
University of Texas at Arlington
Arlington, TX 76019-0437
U.S.A.
snaras@uta.edu

Kenneth H. Price
College of Business Administration
University of Texas at Arlington
Arlington, TX 76019-0437
U.S.A.
price@uta.edu

Abstract

Extreme programming is currently gaining popularity as an alternate software development methodology. Pair programming, a core practice of this methodology, involves two programmers working collaboratively to develop software. This study examines the efficacy of pair programming by comparing the performance, effectiveness, and efficiency of collaborative pairs with those of individual programmers treated as control groups. In a controlled laboratory experiment involving student subjects, pairs for every level of programming ability, to an average level, two factors were manipulated: programming setting (collaborative pairs versus individual) and programming task complexity (high versus low). Participants who worked in the individual condition were randomly assigned to individual pairs. The performance and efficiency of the collaborative pairs were then compared with those of the best performer and the second-best performer of each nominal pair. Results indicated that programming pairs performed at the level above the second-best performer and at the level of the best performer in a group setting. This relationship was found to be consistent across all levels of task complexity. Consequently, there was evidence of an “assembly line effect,” where the performance of a collaborative pair exceeds the performance of its best member working alone. While this finding may appear counterintuitive due to the general perception of two heads being better than one, it is consistent with the findings of small group research. When affective responses were considered, programming pairs reported higher levels of satisfaction than those of the best and second-best performing members in nominal pairs. They also showed higher levels of confidence in their performance.
public void start() { }

Rule 4

public void start() (lines of Java code here)

This is the way that a method is defined, and the start() method is first defined or written in the Applet class. When the programmer uses it and puts lines of code between the braces, that is called overriding the method. Override means to use again and to add lines of code to the original definition.

You know that start() is classified as a method because it has parentheses at the end. Even if the method does not receive any argument, it still must be written with parentheses that are empty. The method name, start, begins with a lowercase letter. That tells you that the method is like a function or a subroutine. It is not a constructor method, and it would never be used with the new keyword to construct an instance of a class.

The keyword void means that a call to the method does not return a value. The keyword void is written with all lowercase letters. That is the convention. The keyword public is an access modifier. It means that the method could be called from inside an instance of any other class. The keyword public is written with all lowercase letters. Both public and void are classified as a keyword.

http://userpages.umbc.edu/~emurian/learnJava/swing/tutor/v2/rules/Tutor.html
Table 2: An example of a rule test question across the two types of assessments. The underlying principle required to solve the problem is identical, and the principle was emphasized in the brief rule tutorial.

<table>
<thead>
<tr>
<th>Pre-Tutor, Post-Tutor, and Quiz</th>
<th>Brief Rule Tutorial and Interteaching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which of the following lines would most likely add a JScrollPane object to a JPanel object?</td>
<td>Which of the following lines would most likely add a JList object to a JPanel object?</td>
</tr>
<tr>
<td>a. JPanel.add(JScrollPane);</td>
<td>a. myBigJPanel5.add(JList);</td>
</tr>
<tr>
<td>b. JPanel.add(myJScrollPane);</td>
<td>b. myBigJPanel5.add(myLittleJList1);</td>
</tr>
<tr>
<td>c. myJPanel.add(JScrollPane);</td>
<td>c. JPanel.add(myLittleJList1);</td>
</tr>
<tr>
<td>d. JScrollPane.add(JPanelObject);</td>
<td>d. JList.add(JPanelObject);</td>
</tr>
<tr>
<td>e. myJPanel2.add(myJScrollPane1);</td>
<td>e. JPanel.add(JList);</td>
</tr>
</tbody>
</table>
Interacting Is a Collaboration Session with Two Students Participating

Interacting Objectives

Before the next class meets, you must complete the brief Rules Tutorial for credit (20 points). The tutorial may take 30 minutes to complete. You may repeat the tutorial as often as you want. The link to the Rules Tutorial is given in the Assignments folder on the course B site.

The below questions may appear on the next quiz. The questions embedded in the Java tutor are also eligible to appear on the next quiz.

You should understand the components of the below program at a level given in the Java Tutor. Also read the material posted in Unit 1 and Unit 2 (1-4) of the online course material.

You should prepare for the interacting session to discuss these components with the intention to understand the specific item and any general principle that is reflected in an item or collection of items. An example of a general principle would be to begin the name of a class with a capital letter.

```java
import javax.swing.JLabel;
import javax.swing.JApplet;
import java.awt.Color;
public class MyProgram extends JApplet {
  JLabel myLabel;
  public void init() {
    myLabel = new JLabel("This is my first program.");
    getContentPane().setBackground(Color.YELLOW);
    getContentPane().add(myLabel);
  }
}
```

You should be able to answer the following questions:

1. What is a class?
2. What is a statement? Give an example.
3. What is a separator? Give an example.
4. What is an operator? Give an example.
5. What is a keyword? Give an example.
6. What is an identifier?
7. What does it mean that methods may be inherited from a superclass?
8. What is the meaning of override?
9. How can you identify a series of characters as the name of a method?
10. What is a constructor method? What properties of the syntax make it a constructor method?
11. Describe the position and functions of the terms in a statement that uses a method to change a property of an object.

During the interacting session, you may have access to the explanations of items that were presented in the tutor:

http://userpages.umbc.edu/~emurian/learnJava/swing/tutor/v2/explanations/Explanations.html

You also may have access to the explanations of the rules in the brief tutorial:

http://userpages.umbc.edu/~emurian/learnJava/swing/tutor/v2/rules/explanations/
Interacting Report #1

Date: 
Name: 
Your partner's name: 
If you have questions about the below material during your discussion, post them in the Discussion Board area.

Interacting Objectives

The below questions may appear on the next quiz. The questions embedded in the Java tutor are also eligible to appear on the next quiz.

You should understand the component of the below program at a level given in the Java Tutor.

Discuss the program with your partner with the intention to understand the specific item and any general principle that is reflected in an item or collection of items. An example of a general principle would be to begin the name of a class with a capital letter.

import java.awt.JApplet;
import java.awt.Label;
import java.awt.Color;
public class MyProgram extends JApplet {
    Label myLabel;
    public void init() {
        myLabel = new JLabel("This is my first program.");
        getContentPane().setBackground(Color.YELLOW);
        getContentPane().add(myLabel);
    }
}

You should be able to answer the following questions:

1. What is a class?
2. What is a statement? Give an example.
3. What is a separator? Give an example.
4. What is an operator? Give an example.
5. What is a keyword? Give an example.
6. What is an identifier?
7. What does it mean that methods may be inherited from a superclass?
8. What is the meaning of override?
9. How can you identify a series of characters as the name of a method?
10. What is a constructor method? What properties of the syntax make it a constructor method?
11. Describe disposition and functions of the terms that make a statement that uses a method to change a property of an object.

During the interacting, you may use the explanations of items and rules that were presented in the tutors:

http://userpages.umbc.edu/~emurray/earnJava/swing/tutor/v2/explanations/Explanations.html
http://userpages.umbc.edu/~emurray/earnJava/swing/tutor/v2/rules/vi/explanations/

You may discuss the below questions with your partner during the interacting discussion. The multiple-choice questions are eligible to appear on the next quiz. Your answers here do not have to be the same for each partner, in case you disagree.

Please circle the correct answer for the below multiple-choice questions. Circle the best choice that you can at this point in your learning.

1. Which of the following lines most likely would be used to create a shorthand notation for the compiler to locate the JFrame class, which is built-in to Java?
   a. import JClass/Frame;
   b. access JFrame.class;
   c. import java.awt.JFrame.class;
   d. append java.awt.JFrame;
   e. import java.awt.JFrame;

   How confident are you that you selected the correct answer? 
   Not at all confident: 1 2 3 4 5 6 7 8 9 10 Totally confident.

2. Which one of the following lines most likely would be used to create a shorthand notation for the compiler to locate the JScrollPane class, which is built-in to Java?
   a. import JScrollPane;
   b. import javax.swing JScrollPane;
   c. access JScrollPane.class;
   d. import java.awt JScrollPane;
   e. append javax.swing JScrollPane;

   How confident are you that you selected the correct answer? 
   Not at all confident: 1 2 3 4 5 6 7 8 9 10 Totally confident.
3. Which of the following lines most likely would be used to add a JCheckBox object to a content pane?

   a. getContentPane().Add(myJCheckBox);
   b. container.Add(JCheckBox.Object);
   c. add(container.JCheckBox);
   d. getContentPane().add(myBox);
   e. Add(myJCheckBox);

Enter a letter here:

How confident are you that you selected the correct answer?
Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
Enter a number here:
Select the best answer below by clicking a button.

Which of the following lines most likely would be used to add a JPanel object to a content pane?

- `getContentPane().add(myPanel);`
- `getContentPane().Add(myPanel3);`
- `container.Add(JPanel.Object);`
- `add(container.JPanel);`

Type the Java here, and press Enter:
Answer at the end of the session.
How effective was this interteaching session in helping you to learn the material?

1 = Not at all effective. The session did not contribute to my learning of the material.
10 = Totally effective. The session contributed to my learning of the material.

(Not effective)  1  2  3  4  5  6  7  8  9  10  (Totally effective)

Enter one number that describes the effectiveness for you: ____.

Answer at the end of the session.
How confident are you that you could answer all questions correctly if you were tested on this program right now?

1 = Not at all confident. I could not answer any question correctly.
10 = Totally confident. I could answer all the questions correctly.

(Not confident)  1  2  3  4  5  6  7  8  9  10  (Totally confident)

Enter one number that describes your confidence: ____.
Question 4
How confident are you that you can use the following symbol now to write a Java program?
JApplet
Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
Enter a number here:

Question 5
How confident are you that you can use the following symbol now to write a Java program?
JLabel
Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
Enter a number here:

Question 6
How confident are you that you can use the following symbol now to write a Java program?
MyProgram
Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
Enter a number here:
Interteachers in Action
At M.I.T., Large Lectures Are Going the Way of the Blackboard

The Massachusetts Institute of Technology has changed the way it offers some introductory classes. Prof. Gabriella Sciolla at a class on electricity and magnetism.

By SARA RIMER
Published: January 12, 2009

Equivalence Relations
Procedure

• Fall 2007 (2.5 hr Class)
  – Class 1
    • Pre-Tutor Questionnaires
    • Programmed Instruction Tutor
    • Post-Tutor Questionnaires
  – Homework
    • Prepare for Interteaching
  – Class 2
    • Lecture
    • Interteaching
      – Questionnaires
  – Class 3
    • Quiz
      – Includes Rule Test Questions from the Questionnaires

• Spring 2008 (2.5 hr Class)
  – Class 1
    • Pre-Tutor Questionnaires
    • Programmed Instruction Tutor
    • Post-Tutor Questionnaires
  – Homework
    • Brief Rule Tutor
    • Prepare for Interteaching
  – Class 2
    • Lecture
    • Interteaching
      – Questionnaires (Brief Tutor)
  – Class 3
    • Quiz
      – Includes Rule Test Questions from the Class 1 Questionnaires

http://userpages.umbc.edu/~emurian/2008study/
A Mann Whitney U test was marginally significant for reported Java programming experience between the two classes ($Z = -1.933, p = 0.053$).
Tutor Evaluation

1 = Totally Negative ... 10 = Totally Positive

Scale
- Overall
- Learning
- Java
- Usability

Rating

Fall 2007 (n = 17)    Spring 2008 (n = 16)

Class
Software Self-Efficacy

1 = No Confidence ... 10 = Total Confidence

Occasion
- Pre-Tutor
- Post-Tutor
- Interteaching

Class
- Fall 2007 (n = 17)
- Spring 2008 (n = 16)
Interteaching Evaluation

Interteaching Evaluation: Fall 2007
1 = Not Effective ... 10 = Totally Effective

Interteaching Evaluation: Spring 2008
1 = Not Effective ... 10 = Totally Effective
Multiple-Choice Test Errors: Tutor → Quiz

**Fall 2007**
- **Items:** $r = 0.551$, $p = 0.022$
- **Rows:** $r = 0.039$, $p = 0.881$
- Sample size: $n = 17$

**Spring 2008**
- **Items:** $r = 0.837$, $p = 0.000$
- **Rows:** $r = 0.649$, $p = 0.007$
- Sample size: $n = 16$
Correct Rule Test Answers: Individuals

Correct Rule Test Answers: Fall 2007

Correct Rule Test Answers: Spring 2008
Rule Test Errors

Rule Test Errors: Fall 2007 (n = 17)

Rule Test Errors: Spring 2008 (n = 16)
Self-Reports of Confidence in Answers

Confidence in Rule Test Answers: Fall 2007

Confidence in Rule Test Answers: Spring 2008

1 = No Confidence ... 10 = Total Confidence
4. Which of the following lines most likely overrides a method that is contained in the Applet class?

a. public Void stop{} { lines of Java code here }
b. public void Stop(){ lines of Java code here }
c. public void stop() {lines of Java code here }
d. Public Void Stop() ( lines of Java code here )
e. Public void stop() { lines of Java code here }

Enter a letter here:
How confident are you that you selected the correct answer?
Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
Enter a number here:
Equivalences

• public void init() {} == public void stop() {}
• How was \texttt{public void stop() {} } recognized as a valid form for a method when that particular form did not appear in the tutor?
Relationally Framing

Applet class methods
1. `public void init() {}`
2. `public void start() {}`
3. `public void stop() {}`
4. `public void destroy() {}`

Override
- `public void init() {}`
- `public void stop() {}`

Hierarchical Combinatorial Entailment
11. Given the line, public class MyTextArea extends JTextArea {}, which of the following statements is correct?
   a. JTextArea is a subclass of MyTextArea.
   b. MyTextArea is a subclass of the extends class.
   c. MyTextArea is a superclass of MyTextArea.
   d. JTextArea is a subclass of the JText class.
   e. JTextArea is a class of MyTextArea.

   How confident are you that you selected the correct answer?
   Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.

12. Given the line, public class MyJButton extends JButton {}, which of the following statements is correct?
   a. JButton is a superclass of MyJButton.
   b. JButton is a subclass of MyJButton.
   c. JButton is a superclass of the extends class.
   d. JButton is a class of MyJButton.
   a. MyJButton is a subclass of the Button class.

   How confident are you that you selected the correct answer?
   Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.

13. Which one of the below lines declares myList as a potential instance of the JList class?
   a. myList JList;
   b. JList myList;
   c. myList JList;
   d. myList JList;
   e. JList myList.

   How confident are you that you selected the correct answer?
   Not at all confident. 1 2 3 4 5 6 7 8 9 10 Totally confident.
17. Which of the following lines would most likely add a JScrollPane object to a JPanel object?
   a. myJPanel.add(my JScrollPane1);
   b. JComponent.add(JScrollPane);
   c. JComponent.add(my JScrollPane);
   d. my JPanel.add( JScrollPane);
   e. JScrollPane.add(my JPanel);

How confident are you that you selected the correct answer?
Not at all confident: 1 2 3 4 5 6 7 8 9 10 Totally confident.

18. Which of the following lines would most likely add a JList object to a JPanel object?
   a. myBigJPanel.add(JList);
   b. myBigJPanel.add(my LittleJList);
   c. JPanel.add(my LittleJList);
   d. JList.add(JPanelObject);
   e. JPanel.add(JList);

How confident are you that you selected the correct answer?
Not at all confident: 1 2 3 4 5 6 7 8 9 10 Totally confident.

19. A Java JApplet program has two methods written in the class. The methods are not nested. What is the total number of braces, { and } added together, that are needed for this program.
   a. 9
   b. 6
   c. 3
   d. 4
   e. 2

How confident are you that you selected the correct answer?
Not at all confident: 1 2 3 4 5 6 7 8 9 10 Totally confident.
Rule Test Errors: Fall 2008 (n = 8)

Rule Test Errors: Fall 2008

Occasion

Pre-Tutor  Post-Tutor  Rules Tutor  Interteaching  Quiz

Rules Tutor: Identical Multiple Choice Questions
Correct Rule Test Answers: Fall 2007

Correct Rule Test Answers: Spring 2008

\[ r = 0.98, n = 8, p = .000 \]

\[ r = 0.87, n = 9, p = .002 \]

Pre-Tutor, Post-Tutor, Interteaching, Quiz
History Counts

Fall 2007 (n = 17)

Spring 2008 (n = 16)
History Counts

Fall 2007 (n = 17)

Pre-Tutor Rule Errors vs. Pre-Tutor Software Self-Efficacy

R Sq Linear = 0.501

Spring 2008 (n = 16)

Pre-Tutor Rule Errors vs. Pre-Tutor Software Self-Efficacy

R Sq Linear = 0.588
History Counts

Fall 2007 (n = 17)

Spring 2008 (n = 16)
History Counts

Fall 2007 (n = 17)

R Sq Linear = 0.018

Spring 2008 (n = 16)

R Sq Linear = 0.285
History Counts

Fall 2007 (n = 17)

R Sq Linear = 0.002

Spring 2008 (n = 16)

R Sq Linear = 0.405
Classification: All… or Nothing?

The below questions are based on the design of the Java programming language and associated conventions of the language. **Classification** refers to keyword, class, object, method, separator, and operator. Give the most informed rating that you can at this point in your understanding of Java.

1. How similar to each other are the following two items in terms of classification?
   - (1) import
   - (2) new
   **Classification:** Not Similar 1 2 3 4 5 6 7 8 9 10 Highly Similar
   **Enter a number here:**

2. How similar to each other are the following two items in terms of classification?
   - (1) myLabel
   - (2) JLabel
   **Classification:** Not Similar 1 2 3 4 5 6 7 8 9 10 Highly Similar
   **Enter a number here:**

3. How similar to each other are the following two items in terms of classification?
   - (1) getContentPane()
   - (2) init()
   **Classification:** Not Similar 1 2 3 4 5 6 7 8 9 10 Highly Similar
   **Enter a number here:**
Classify: (1) import  (2) new

1 = Not Similar … 10 = Highly Similar

Occasion
- Pre-Tutor
- Post-Tutor
- Interteaching

Rating

Class
- Fall 2007 (n = 17)
- Spring 2008 (n = 16)
Classify: (1) myLabel  (2) JLabel

1 = Not Similar … 10 = Highly Similar

Class

Fall 2007 (n = 17)  Spring 2008 (n = 16)

Occasion

- Pre-Tutor
- Post-Tutor
- Interteaching
Classify: (1) `getContentPane()`  (2) `init()`

1 = Not Similar … 10 = Highly Similar

![Box plot showing rating by occasion and class]

Occasion
- Pre-Tutor
- Post-Tutor
- Interteaching

Class
- Fall 2007 (n = 17)
- Fall 2008 (n = 16)
Classify: (1) MyProgram  (2) JApplet

1 = Not Similar ... 10 = Highly Similar

Occasion
Pre-Tutor  Post-Tutor  Interteaching

Rating
7  8  9  10

Class
Fall 2007 (n = 17)  Spring 2008 (n = 16)
Challenges with Programmed Instruction

• It is labor intensive to develop.
  – We have proposed to develop a generic shell.
• There are conceptual issues regarding the size of a learn unit.
  – The opportunity for repetition, until a multiple-choice item is answered correctly, can lead to careless reading.
Challenges with Interteaching

- A rare student will show an aversion to collaborative learning.
- Pairs of students need different amounts of time.
- It is difficult to assess the “quality” of a collaboration objectively.
- “Understanding” is more than an intraverbal performance.
Challenges with Lecture

• I have to know what I’m talking about.
Conclusions

1. Programmed instruction is an effective tool in technology education.
   - It meets the needs of the individual learner.
   - The instructional design can promote meaningful learning and self-confidence.
   - The tutoring system is well-received by novitiate learners.

2. Interteaching may add value, but there are issues of retention and transfer.

3. The competency attained sets the occasion for advanced learning with enthusiasm.

4. Students like the tutor and the interteaching, and so do I.

5. I also like to use lectures with hands-on learning and classroom collaboration among students.
PRACTICE MAKES PERFECT!!

Questions?

The tutor, the source code, and all instructional material are freely available on the web.