Instructional Technology in the CS Introductory Programming Classes[1]

Nancy E. Miller and Donna S. Reese[2]

Abstract

Many students with widely varying backgrounds are required to take an introductory programming course in Computer Science. Mississippi State University has three courses in introductory programming that teach these concepts and differ only in the programming language that they use. Currently these courses are taught in a lecture only format with programming assignments required as part of the students' grades. Many of these introductory programming students struggle with the concepts in these classes. The wide availability of the World Wide Web to these students provides us with an opportunity to develop supplemental course material for these students to help demonstrate the basic concepts, and advance the students' capabilities in developing computer programs to solve a variety of problems. This paper discusses the development of the instructional material to provide asynchronous learning activities that will be available to the students, both in the lab and outside the lab/lecture time frame.

Introduction

Introductory programming courses in Computer Science are required by most of the engineering disciplines, business information systems, math education, and industrial technology at Mississippi State. The number of students enrolled in this set of classes in the 1997 calendar year was approximately 400 students across these disciplines. For many of these majors, this is the only course in computer programming that they are required to take. For Computer Science and Computer Engineering majors, it provides the firm foundation in programming that they build on in the rest of their academic program. The three introductory programming courses differ only in the programming language used (CS 1253 – Pascal, CS 1233 – C, and CS 1213 – FORTRAN, hereafter we will refer to these courses collectively as CS 12X3).

Table 1 shows the enrollment in CS 12X3 over the last two academic years. As can be seen from this data, more than half of the students in these classes are from some engineering major. The college of engineering at Mississippi State University is beginning a computer initiative in the fall of 1999 that will require all incoming freshman engineering students to own a computer. Further motivation for this study is to develop instructional material that can be used in the classroom when these computers come to campus. In addition to it being required of most majors, the CS 12X3 course is listed in the illustrative schedule in the first semester of the freshman year for computer science, computer engineering and electrical engineering majors and in the first semester of the sophomore year for industrial and civil engineering majors. Since the computer initiative will begin with freshmen students, this course is at an appropriate place in the curriculum for us to make an early impact.

---

[1] This work funded through a Schillig Teaching Grant and funding from the Robert M. Hearin Foundation.

[2] Mississippi State University, Computer Science Department, Box 9637, MSU, MS, 39762.

Approach

For this project we have attempted to provide two different kinds of instructional support. The first involves the development of Java applets to illustrate the programming language constructs. The second is the incorporation of a courseware package to provide cooperative learning environments and asynchronous assessment activities. In addition, we are attempting to address the students' perceptions that this class is not useful in their major with the use of engineering application oriented lab activities.

Java Applet

We began this project with a search of the web for animation applets that were already developed. Most of the work that we found was aimed at the CS II level and above, and very little seemed to be available for introductory programming concepts. A summary of the animations that were found is available at URL http://www.cs.msstate.edu/~ng/dis/mydis.html.

Major	Fall 1996	Spr. 1997	Sum 1997	Fall 1997	Spr. 1998	Sum 1998	Fall 1998
ASE	1	0	0	2	0	2	4
BE	15	4	3	13	8	6	12
CE	6	3	8	16	29	2	29
CHE	2	0	0	0	0	5	2
CPE	32	30	2	47	29	2	70
CS	24	14	3	31	30	5	50
EE	36	33	6	27	28	3	41
IE	19	17	1	11	8	9	11
ME	6	4	1	4	1	0	3
Total Eng	141	105	24	151	134	34	222
Other Majors	110	69	27	69	67	17	71
Total Students	251	174	51	220	201	51	293

Table 1 - Enrollment Counts for CS 12x3

During the spring and summer of 1998, we took one of the sample applets that we found, which demonstrated sorting algorithms (http://www.cai.j.dendai.ac.jp/Anim-Sort/Animation.html), and used it as a framework for developing introductory programming animations. Table 2 shows a list of the programming concepts covered by these animations. Students are shown this applet in the lecture and then encouraged to pursue the use of it on their own. The animations include the same set of programs in all three languages (where possible).

Control buttons on the applet allow the student to watch the execution of each example at varying speeds, or to single-step through the programs. Individual windows indicate the variables involved in the program with their values (if defined). In addition, an input window shows the current status of input by highlighting what is to be read. The output window shows all output produced by the program through the current point of execution. Each line of the program is highlighted as it is executed, allowing students to follow through the basic programming constructs. Users may pick the language to be viewed, as well as the particular example and speed of its execution. Figure 1 is an example snapshot of a program during execution, with the current line being executed shown as highlighted, and the input and output produced so far.

Sample Program	**Topics Covered**
Beginning	Basic hello world program
Calculation	Simple arithmetic
If-else	If-then-else statements
Nested	Nested If-then-else statements
Booleans	Compound conditions
Case	Case statements
End of File	Simple I/O with EOF loop
Counting	Summing numbers with while
For loop	Simple for/do loops
Sentinel	Counting loop with early exit
Procedure	Simple procedures, no arguments
Function	Simple function
Parameters	Procedures and functions with arguments
Nested functions	Functions calling other functions
1D arrays	Simple 1D arrays with max
Searching	Simple linear search
2D arrays	Two dimensional arrays
Recursion	Basic direct recursion
Text files	Simple file I/O
Records	Simple structures/records

Table 2 - Sample programs covered in Java applet

Courseware

In addition to the Java applets, we would like to support cooperative and asynchronous learning. We researched the currently available courseware for a package that would allow web-based access, and that would support online assessment activities (Brusilovsky 1998, Bethoney 1997, Freeman and Ryan 1998). The three packages that seemed to address both of these issues were TopClass, Learning Space from Lotus (Learning Space 1998), and WebCT. A decision by MSU to support a site license for WebCT made use of that package the most attractive.

This courseware system also supports online assessment. Students' mastery of this assessment material is recorded and tracked by the courseware system. In some cases, students can get automatic feedback on their level of mastery of this material. Instructors can track the topic areas that students are struggling with, and see how individual students are progressing in each of the areas.

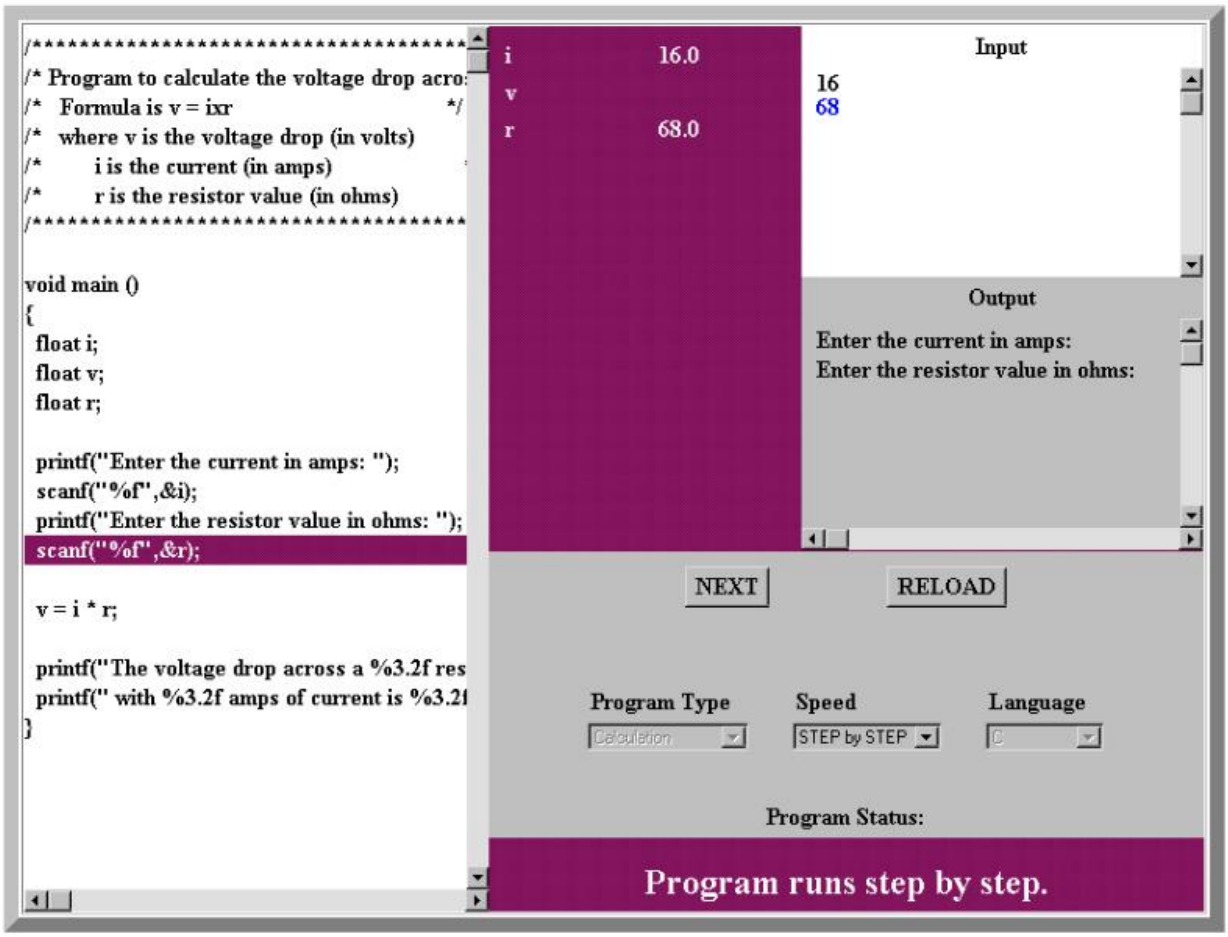

Figure 1 - Algorithm Animation Snapshot

The courseware package also includes support for discussion groups. These groups can include the instructor or other students in the class, and can be organized by topic area. Students learn from other students' questions, and from discussions that provide additional examples and material to reinforce the course topic areas.

This courseware system also supports the inclusion of information about the students in the course. This material can include visual and textual material describing each student's interests and expectations for the course. This information helps students to feel more comfortable with each other, and helps foster the cooperative learning environment for the course.

We plan to use WebCT to provide asynchronous learning activities that will be available to the students both in the lab and outside the lab/lecture time frame. Since most students own their own computers, these learning activities will be available with any web connection at any hour of the day or night the students wish to study. These learning activities will include the presentation of additional example problems for the students to work, animations of the different programming constructs, and self-assessment activities.

While access to WebCT was not available in the fall semester, an individual web page for each student in the class is used. This page includes a picture of the student, their major and a link for sending email. We have also included the class instructor and graders on these pages. This has helped the students get to know each other in addition to helping the instructors more easily learn the names of the students.

Discipline-specific Labs

In addition to the instructional technology for these classes, we are modifying the programming assignments in this class to allow for more relevance to each student's major. We are working with each of the engineering departments to identify a problem in the domain for the final programming assignment of the semester. The final lab assignment for this class normally involves a problem requiring the use of one-dimensional arrays and functions. We realize that the students will have had few, if any, classes in their majors. These assignments were chosen to introduce the students to a problem in their major that demonstrates the programming applicability without requiring significant knowledge in the field. These problems do assume the college algebra prerequisite in math.

As an example of the type of problem that was used, electrical engineering students implemented a simple 1-dimensional problem to construct a resistor for a given resistance from a list of resistor denominations. This problem is solved by summing the reciprocals of the resistor denominations to achieve the desired resistance. Aerospace engineering students implemented a simple 1-dimensional problem to use data collected during a wind tunnel test on an aircraft wing to interpolate the coefficient of lift for a particular attack angle. This problem is solved with linear interpolation. Both of these problems provide a fairly simple programming assignment that makes use of 1-dimensional arrays and functions. Similar problems in the other disciplines were assigned.

Assessment

The Computer Science Department has already developed a programming proficiency exam that is used to test the students' readiness to enter the Computer Science I class (which follows CS 12X3). For purposes of measuring the educational impact of this instructional technology on these students, this exam was administered to the spring 1998 CS 12X3 classes to provide a baseline measure of the success of this class. We will again administer the proficiency exam to the students in the CS 12X3 classes in fall 1998 and spring 1999 who are exposed to the additional instructional material. This will give an indication of whether this additional instructional material is helping to improve the students' learning.

In addition to the educational impact, we would like to measure the impact on the students' perceptions of this course. We currently hear many comments about the level of difficulty in these classes. We have developed a ten-question survey that was added to the proficiency exam to allow the students to indicate the perceived level of difficulty in the class and their general attitude towards computer programming. Again, we have administered this survey to the spring 1998 students to provide a baseline for comparison with the students in the future that take the course that makes use of the enhanced instructional technology.

Results

Results of the survey and proficiency exam from the first experimental group in the fall 1998 were discouraging. Because WebCT was not available in the fall semester, there was no tracking of the students to see who used the animations. The survey questions asked the students about the entire class (including lecture, labs, and animations). All of the students in the CS 12X3 classes felt the class helped make programming easier for them. Computer science and computer engineering majors in the CS 12X3 classes felt that the class helped them feel more comfortable writing a program to solve a problem in their major. Since different instructors taught the CS 12X3 classes (different by languages), we analyzed the survey questions by language as well. All students enrolled in CS 1233 (using the C language) felt the class helped make programming easier and they could see applications to their major. In the FORTRAN class (CS 1213), only the engineering majors (excluding computer science and computer engineering) felt this class helped them feel more comfortable in solving a problem in their major. The computer science, computer engineering, and other engineering majors taking the Pascal class (CS 1253) felt this class applied to problem solving in their major. Because each language has a different instructor, we are not able to determine if differences are due to the instructor or the programming language that is taught.

Future

During the spring of 1999 we will be loading the animations and quizzes into WebCT so we will have tracking of the students. We will also have the same instructor for multiple sections as well as a control group (one section that does not have access to WebCT) to better analyze the survey results. We will also improve the survey questions to ask the students about not only the class, but also the various components of the class (lecture, laboratory, animations, and online quizzes). During the summer of 1999 we will run a workshop for engineering faculty on the use of these tools. We anticipate being able to teach 2-3 faculty members from each department in a one-week short-course. This course will cover the use of the WebCT product, as well as our experiences with different types of online animations and assessment activities.

During the 1999-2000 academic year we will have access to the first set of students who will have their own computers. The computer science students in the CS 12X3 classes will have laptops that can be brought to class. During this academic year we will investigate experiences that can be achieved through the use of these laptops in the classroom. We recognize that only a small number of students will have access to laptops, and we will not require their use (except perhaps as a laptop only section). However, we do want to begin to explore ways in which the use of these machines by students in the classroom can benefit the educational process.

Acknowledgements

Robson Ng helped with the web search for the initial algorithm animations. Jill Tolbert has integrated all of the examples into the animation framework.

References

Brusilovsky, Peter (1998) “Integrating Hypermedia and Intelligent Tutoring Technologies: From Systems to Authoring Tools,” http://www.pitt.edu/~al/aied/ brusilov.html, (Accessed 27 January 1998).

Bethoney, Herb (1997) “Computer-based Training on the Web,” http://www8.zdnet.com/pcweek/ reviews/0818/18ibt.html, (Accessed 9 February 1998).

Freeman, Howard and Steve Ryan (1998) “Webmapping: Planning, Structuring and Delivering Courseware on the Internet,” http://westworld.dmu.ac.uk/ webmapper/webmapper.html, (Accessed 9 February 1998).

Learning Space (1998) http://www.lotus.com/home.nsf/ welcome/learnspace, (Accessed 9 February 1998).

Nancy E. Miller

Nancy E. Miller is an Associate Professor of Computer Science at Mississippi State University. She holds a bachelor's degree in Mathematics from Northwest Missouri State University, a master's degree in Computer Science from Iowa State University and a Ph.D. from Iowa State University in Higher Education. Her dissertation was entitled "Team Programming in an Undergraduate COBOL course." She has been teaching computer science at the university level since 1977. Professor Miller's teaching experience is captured in undergraduate-level texts on file structures available in Pascal and Ada (File Structures using Pascal and File Structures with Ada). She is the author of several articles dealing with computer science education, particularly in teaching beginning undergraduate courses.

Donna S. Reese

Donna S. Reese is an Associate Professor and Undergraduate Coordinator of Computer Science at Mississippi State University. She holds a BS degree in Computer Science from Louisiana Tech University and an MS and Ph.D. in Computer Science from Texas A&M University. She has been teaching computer science at the university level since 1989. Dr. Reese's research interests include parallel and distributed computing, object-oriented programming and instructional technology.