Interactive Student Support for Introductory Computer Science Courses

Barb Ericson and Erika Rogers

Abstract

This paper describes the development of interactive multimedia modules which provide supplementary tutorials on basic topics covered in introductory computer science classes. Two such modules were implemented on the topics of number conversions and basic operating system information. The first versions of the tutorials were evaluated in a study using protocol collection and a post-test with a number of student participants. Results from this study led to the re-design of several portions of the tutorials, which now have a greater level of interaction and progressive assistance for problem-solving.

Introduction

Increasing emphasis on computer-based technology in the workplace and at home is forcing even relatively young computer science departments to reconsider their curriculum in light of these new demands. However, changing the course content and the supporting infrastructure is often a slow process, and many students are caught in a transition where they are expected to have skills for higher level classes which were not covered under their original program of study. An example of this is here at Clark Atlanta University, where our introductory computer science courses started out using the Pascal language and VMS operating system, and then converted to the C language under a UNIX environment. In order to assist the students in making this transition, a project has been started to develop interactive multimedia modules which provide supplementary tutorials on basic topics covered in the early CS courses. Two topics were chosen from the first freshman level course, Structured Programming in C (CIS 105), based on the instructor's notes and the textbook. A rapid prototyping strategy was adopted in order to be able to get early feedback from the students, and results from an observational study led to a number of improvements for the second version. The following sections describe the two modules developed and highlight features that are felt to be particularly effective.

Overview of Modules

The first two modules consist of a Number Conversion tutorial which covers converting numbers to different bases (especially binary, octal and hex), and an Operating Systems tutorial which introduces files and directories, the program cycle, and basic UNIX commands. Fig. 1 shows the introductory screens for both of the modules. A common "look and feel" interface was maintained across modules, including similar layouts and colors, help screens, error messages, test options, and comment capabilities (both structured and unstructured). Both modules feature a graphical form-based user interface with buttons, file browsers, and images (both static and clickable). The help screen, which automatically pops up when a student chooses a topic, shows an annotated replica of the upcoming interactive window with an example, together with a descriptive information window, as shown in Fig. 2. The tests for each module also have a similar format, with a number of questions, an area for user input, and immediate feedback. If the student types in a wrong answer, a window pops up from the appropriate part of the tutorial, enabling the student to review the corresponding material. Fig. 3 shows an example of this feature for the Number Conversion tutorial. When the supporting window appears, the values from the test question are automatically entered into the problem statement.

Version 1 Design and Protocol Study

While both modules are interactive, the Number Conversion tutorial represents a dynamic problem-solving scenario, while the Operating Systems tutorial presents relatively static information. Therefore, the first version of the project emphasized the use of animations of the conversions between number bases to illustrate the dynamic process involved. The prototype tutorials were

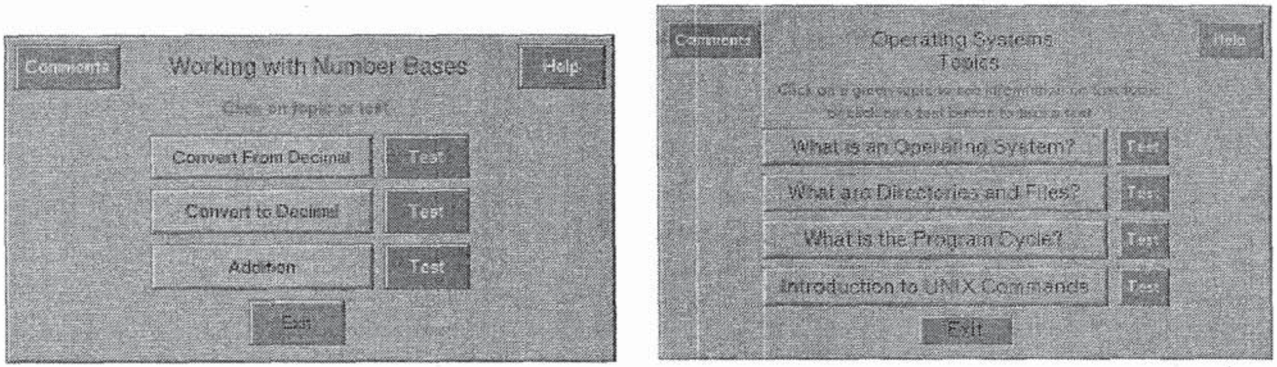
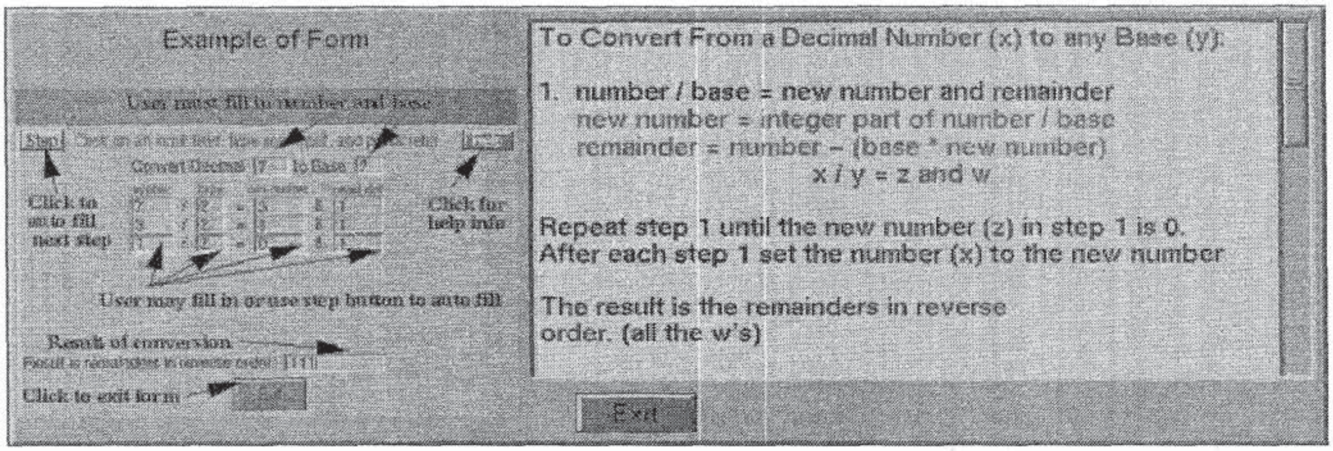


Figure 1: Introductory screens for both modules.


Example of Form
User must fill in number and base
Step
Click to auto fill next step
Click for help info
User may fill in or use step button to auto fill
Result of conversion
Click to exit form
Exit

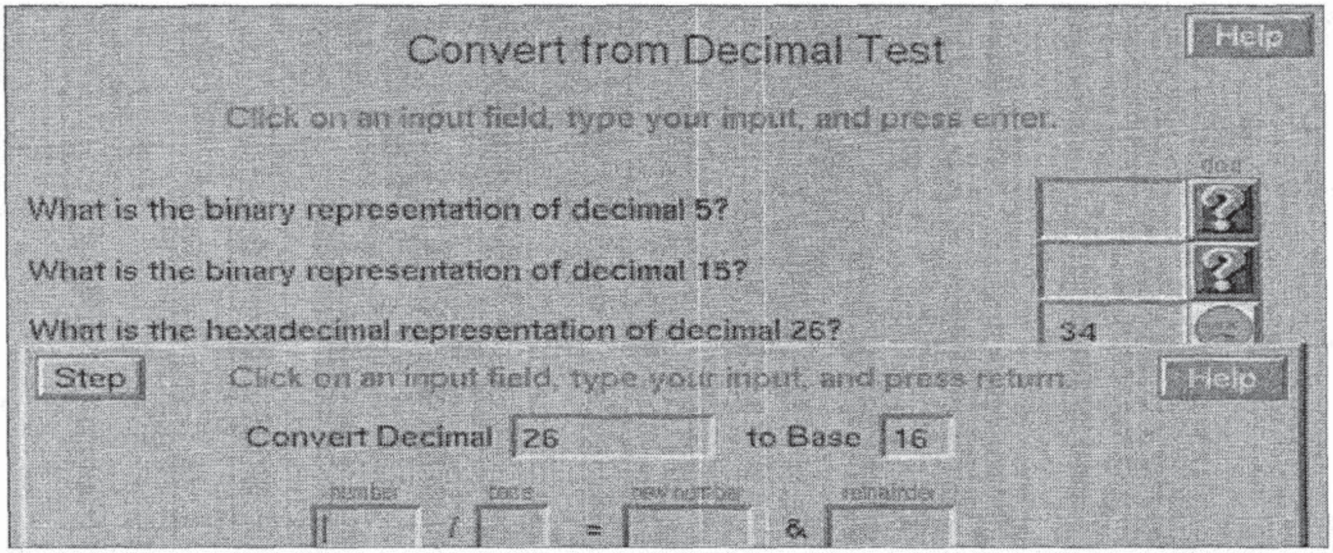
To Convert From a Decimal Number (x) to any Base (y):

1. number / base = new number and remainder
   new number = integer part of number / base
   remainder = number – (base * new number)
   x / y = z and w

Repeat step 1 until the new number (z) in step 1 is 0.
After each step 1 set the number (x) to the new number

The result is the remainders in reverse
order. (all the w's)

Figure 2: Example help screen from Number Conversion Tutorial.



Click on an input field, type your input, and press enter.

What is the binary representation of decimal 5?

What is the binary representation of decimal 15?

What is the hexadecimal representation of decimal 26? 34

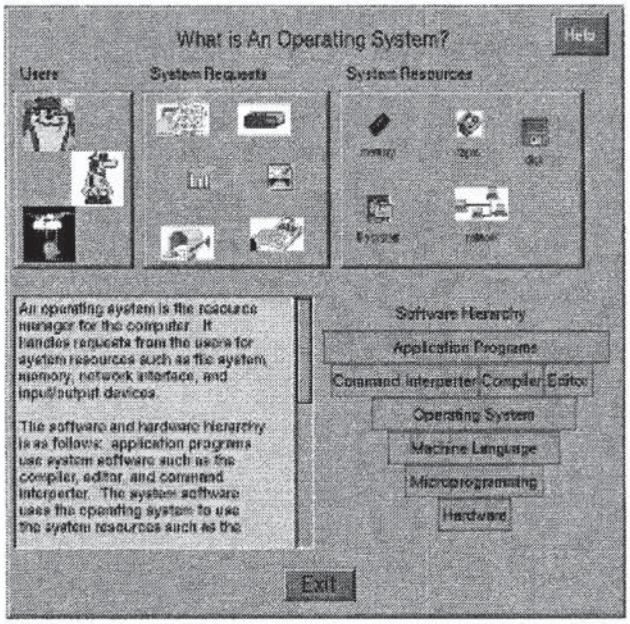
Click on an input field, type your input, and press return.

Convert Decimal 26 to Base 16

Figure 3: Example test screen with incorrect answer.

evaluated in a study using protocol collection and a post-test with nine student participants, primarily drawn from the CIS 105 class and ranging in experience from high- to low-ability (according to their course grades). Each student's interaction with a module was captured in a trace file, and student comments were solicited through the Comments form, as well as through verbal discussion with one of the investigators.

Feedback from the students revealed a number of interesting results and reactions which hadn't been anticipated. Generally, the students liked the courseware and indicated that they could imagine instances when it would be useful for them to use again. They found the Operating System module easy to use and interesting, and liked the fact that the diagrams were similar to those used in class (Figs. 4 and 5). Students also particularly liked the animations of the conversions between number bases, and several of them remarked that they recognized the process from the book. However, these same students didn't even know where to begin to convert a decimal number to hexadecimal when asked to do so in the post-test. There seemed to be no retention of the conversion process and this was felt to be a major flaw in the tutorial. Otherwise, actual negative comments were few in number and were directed at the type size and the UNIX man pages (demonstrated in the Operating System tutorial).



An operating system is the resource manager for the computer. It handles requests from the users for system resources such as the system, memory, network interface, and input/output devices.

The software and hardware hierarchy is as follows: application programs use system software such as the compiler, editor, and command interperter. The system software uses the operating system to use the system resources such as the

Figure 4: What is an Operating System (static pixmaps).

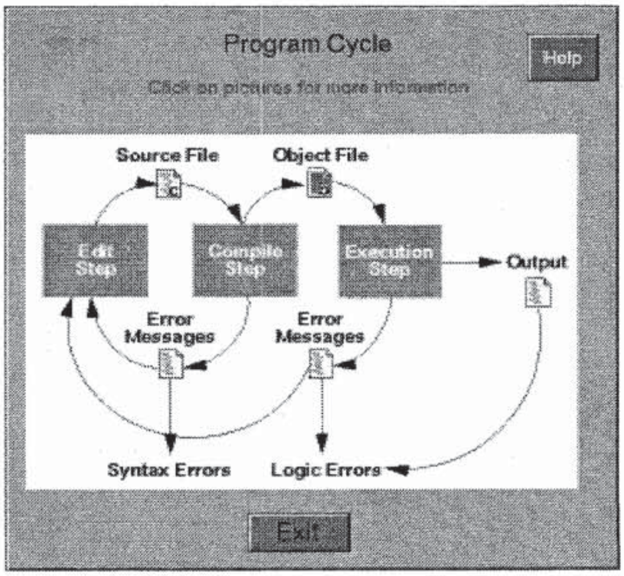
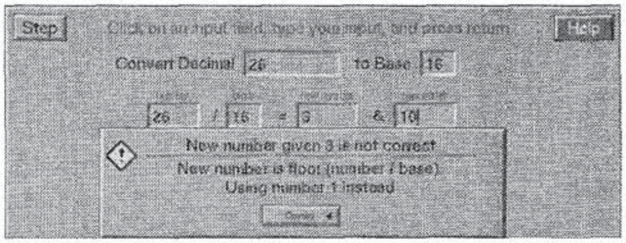


Figure 5: The Program Cycle (clickable pixmaps).

Through observation, we discovered that the students had a number of difficulties: starting the more complicated forms, scrolling the animation window for speed, clicking on an input field before typing, pressing return when finished, and entering numbers. It was felt that some of these problems had to do primarily with the students' lack of experience with computers in general, and windowing systems in particular. On the other hand, some learning difficulties also emerged. Many students were fearful of answering the post-test questions unless they were sure of the answers. They preferred to use the module to find the correct answer. When students thought that they were sure of the answer and found by negative feedback that they had the wrong answer, they wanted to know why. Many students navigated the modules by answering the test questions and only looked at the available information if they got a question wrong. Some students said that they wanted more tests.

Based on these results, the courseware was re-designed to require more student interaction, particularly in the problem-solving module. Instead of watching an animation of base number conversion, the student is now led through the process with levels of support to assist the student. For students with no idea how to begin the process, a help window with an example together with the process steps is displayed. A step button allows the user to let the computer automatically do the next step. If incorrect input is entered, a window appears with an explanation about how to calculate the right answer (Figs. 6 and 7).



Click on an input field, type your input, and press return

Convert Decimal 26 to Base 16

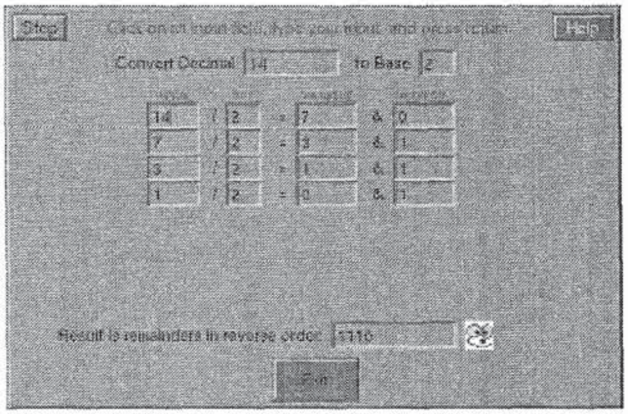
26 / 16 = 3 & 10

New number given 3 is not correct

New number is floor (number / base)

Using number 1 instead

Figure 6: Error assistance in Number Conversion.



Convert Decimal 14 to Base 2

14 / 2 = 7 & 0

7 / 2 = 3 & 1

3 / 2 = 1 & 1

1 / 2 = 0 & 1

Result is remainders in reverse order 1110

Figure 7: Interactive Number Conversion Window.

The modules were also changed to be more test driven. A "do it" or "explain" button was added to the on-line tests that lead the user through the process of finding the correct answer or help explain why their answer was wrong. Since students also indicated that they would like more tests, a facility for adding more tests was incorporated, and a "next test" button on the interface now allows the student to attempt additional tests.

Conclusion

The development of interactive multimedia tutorials for introductory computer science courses can provide important support, not only for incoming freshmen learning the concepts for the first time, but also for more advanced students who may need to review basic principles or learn new system configurations. In this project, two prototype modules were developed and tested with a number of students. This early feedback was extremely valuable in the re-design process, and emphasized the need for a more interactive approach, and the provision of scaffolded assistance. Future work is planned to add more topics as well as to extend the multimedia environment to include sound and video clips, hyperlinks, more clickable pixmaps, and the re-introduction of animations to support a more adaptive interface.

Acknowledgments

The work described in this paper is supported by the National Science Foundation Grant IRI-9502289.