Evaluating the Use of Flowchart-based RAPTOR Programming in CS0

Michael Thompson
Division of Mathematics, Engineering, and Computer Science
Loras College
Dubuque, Iowa 52001
michael.thompson@loras.edu

Abstract

The Introduction to Computer Science, or CS0, course can be challenging to teach effectively. The purposes and pedagogical methods used in this course vary significantly. At Loras, it is a required foundational course for Computer Science and Management Information Systems majors, as well as a requirement for many majors in the Business program. This paper looks at enhancing students' learning experience by analyzing different pedagogical methods and challenges in teaching this course. The primary focus of this study involves comparing the teaching of a flowchart-based programming environment (RAPTOR) with teaching general problem-solving techniques without using specific programming concepts or languages. The results suggest that using flowchart-based programming can be beneficial to students' problem-solving abilities and their ability to learn basic spreadsheet and database concepts with Microsoft Office.

1 Background

It can be a challenge to effectively teach the Introduction to Computer Science, or CS0, course. There are many different perspectives on how to teach non-major level computing courses. Various approaches to these courses were discussed at a panel discussion at the SIGCSE 2010 conference. In this discussion, some of the points included questions regarding whether to focus on problem solving methods without programming, or whether programming components must be included in the course in order for students to understand how to express their ideas and solutions with the precision required by computers [1]. In addition, survey results presented at the SIGCSE 2011 conference demonstrated that the amount of programming content and programming languages used in CS0 courses varied greatly across institutions. The survey demonstrated that there is no clear consensus regarding the amount of programming in CS0, and that it ranges nearly uniformly from very little programming to all programming [2].

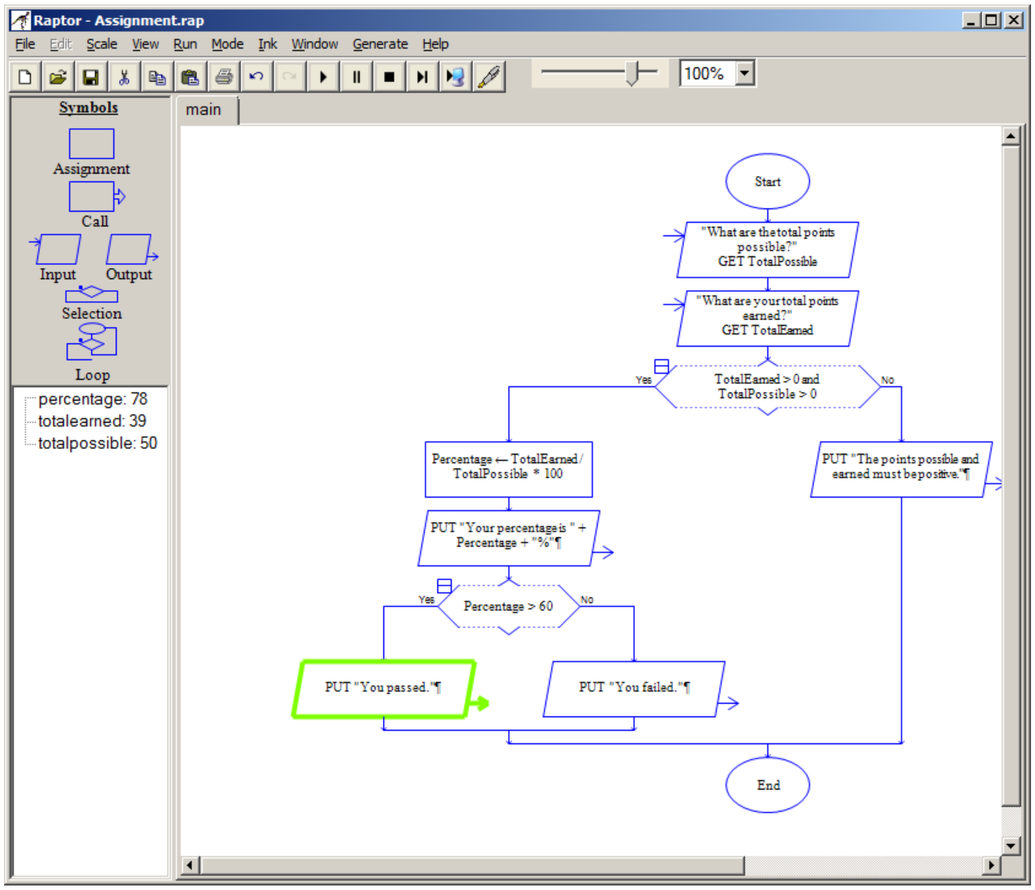
1.1 The CS0 Course at Loras

At Loras, the CS0 course is CIT 110, Computing and Information Technology Basics. It is a required course for both the Computer Science and Management Information Systems majors. The Business program requires it for its majors. The course covers basic databases and spreadsheets in Microsoft Office, as well as general computer concepts relating to hardware and networks. The CS and MIS majors share a common core of computing courses and CIT 110 is a prerequisite for many of the other courses in the core, including courses on databases, networks, and the introduction to programming CS1 course. Furthermore, while the upper-level courses in each major are primarily taught by faculty in the corresponding program, CIT 110 is taught by faculty from both the Computer Science and Management Information Systems programs. Faculty found that the broad range of necessary learning outcomes needed in CIT 110 could not be satisfactorily met over the course of a semester. As a result, faculty from different programs taught the course differently, focusing on the outcomes they felt were most important and ignoring or glossing over other outcomes. Therefore, in the summer of 2010, the CS and MIS faculty decided to replace the programming-related outcomes of the course with an overview of general problem solving techniques so the other outcomes could be met.

When CIT 110 was taught in the fall of 2010, some faculty who taught the course with the new outcomes found that they had extra time at the end of the course. They used this extra time to teach a brief introduction to programming. At the conclusion of the semester, the faculty who did this felt that the inclusion of programming would have helped to better inform their teaching of spreadsheet and database applications in Microsoft Office. Due to this, we decided to experiment with the inclusion of a brief introduction to RAPTOR programming at the beginning of the course in lieu of the problem solving component.

1.2 RAPTOR Programming

RAPTOR is a flowchart-based programming environment that is designed to help students visualize algorithms while limiting the burden of syntax [3]. Loras has been using RAPTOR in its CS1 course for many years as an introduction to basic programming concepts before students are introduced to C++. The RAPTOR interface provides students with an intuitive means of adding sequential, selection, and repetition structures to their flowcharts. In addition, it has an easy to use graphics library that allows students to create interesting programs using animations with relative ease. An example of the RAPTOR user interface while a program is running is shown in Figure 1, and an example of the graphics that can be easily created by a RAPTOR program is shown in Figure 2.


Raptor - Assignment.rap
File Edit Scale View Run Mode Ink Window Generate Help
100%
Symbols
Assignment
Call
Input
Output
Selection
Loop
main
Start
"What are the total points possible?" GET TotalPossible
"What are your total points earned?" GET TotalEarned
Yes
TotalEarned > 0 and TotalPossible > 0
No
Percentage ← TotalEarned / TotalPossible * 100
PUT "The points possible and earned must be positive."¶
PUT "Your percentage is " + Percentage + "%"¶
Yes
Percentage > 60
No
PUT "You passed."¶
PUT "You failed."¶
End

```
percentage: 78
totalearned: 39
totalpossible: 50
```

Figure 1: The RAPTOR Programming Environment.

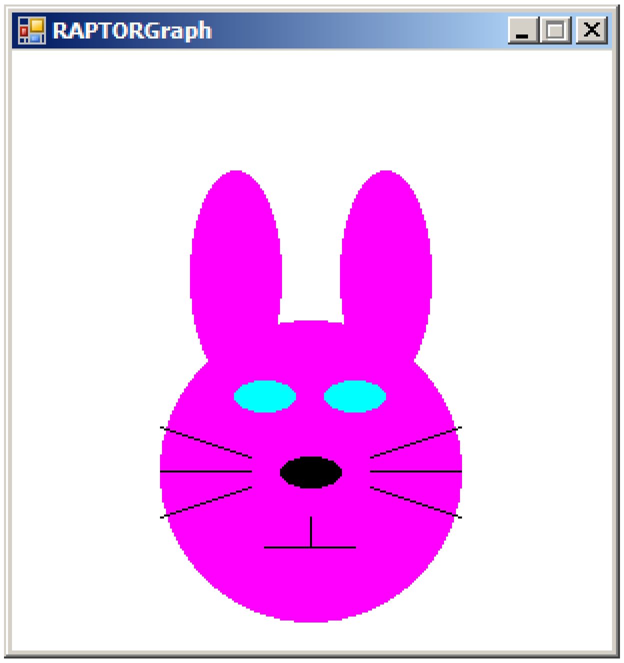


Figure 2: An Example of Graphics in RAPTOR.

2 Programming or Problem Solving?

In spring of 2011, it was decided to teach some sections of CIT 110 beginning with a focus on problem solving and others beginning with an introduction to RAPTOR programming in order to determine whether the perceived benefits of teaching RAPTOR early in the semester would be realized. After the introductory RAPTOR and problem solving units, all sections were taught in parallel, with minor differences based on the instructors' styles. Common assignments and exam questions were developed in order to measure how much students learned about both problem solving and the Microsoft Office applications essential to the learning outcomes of the course.

2.1 Methodology

Three sections of CIT 110 were taught in spring of 2011. During the first two weeks of the course, one section was taught using RAPTOR and two sections were taught using problem solving. Prior to introducing these methods, students in all sections of the course received a survey that was designed to gauge each student's overall interest in Computer Science, as well as their mathematical and computing background. A similar survey was given at the end of the semester in order to determine how student attitudes changed over the course of the semester.

The section teaching RAPTOR introduced students to programming in RAPTOR, making use of its animated graphics functionality. Over this time, this section covered sequential, selection, and repetition program control, as well as the use of variables and function calls.

The sections covering problem solving introduced techniques based on a modified version of the “How to Program It” document [4], based off the ideas in the book “How to Solve It” by G. Polya [5]. The problem solving component focused on the ideas of understanding the problem, designing a solution to the problem, and evaluating the correctness of the solution. When gaining an understanding of the problem, students were taught how to ask questions that led to a deeper knowledge of its specifics. When designing their solutions, students created informal flowcharts written on paper that were designed to both solve the problem and handle any incorrect inputs. When evaluating this solution, students were taught how to define test cases for the purposes of testing how well their proposed solutions solved the problem and used these test cases to evaluate their flowcharts.

Following the two-week introductions to either problem solving or programming in RAPTOR, all sections of the course were run essentially in parallel with some common assignments across sections. For the purposes of addressing the question of whether teaching programming or problem solving would be more beneficial, three specific assignments were chosen. One was the culminating assignment in Microsoft Excel, another was the culminating assignment in Microsoft Access, and the third was a data analysis project that students could choose to complete in either Excel or Access. Each assignment was graded using a common rubric that evaluated how well students used the application-specific techniques required in the assignment. In addition, each assignment was rated based on how well students demonstrated effective problem solving techniques using a problem solving rubric based on a problem solving rubric at the Schreyer Institute for Teaching Excellence at Penn State University [6].

This problem solving rubric focused on evaluating students’ analysis of the problem and translating this analysis to their solution in Access or Excel. For instance, in the sections focusing on problem solving, students were taught to examine the inputs to a problem and find an acceptable range of values on the inputs, while students in the section being taught RAPTOR were not. Then, in the problem solving rubric for Excel, students were expected to identify and handle the range of acceptable values using data Excel’s data validation functionality. Students in all sections were taught data validation in Excel, but were not told how to apply it in the description of the Excel assignment.

2.2 Results

The results of each assignment from the spring of 2011 are shown in the tables below. Table 1 contains the results of the raw scores (out of 100) and Table 2 contains the results of the problem solving rubric (out of 16).

	Excel			Access			Project			Overall	
	PS	R		PS	R		PS	R		PS	R
Mean	82.54	89.27		71.78	81.46		77.16	82.76		77.59	84.60
Std. Dev.	12.26	15.72		20.36	15.45		14.04	11.34		16.21	14.73

Table 1: Score Comparison Between Problem Solving (PS) and RAPTOR (R) Sections.

	Excel			Access			Project			Overall	
	PS	R		PS	R		PS	R		PS	R
Mean	12.39	13.96		10.11	11.75		12.55	13.20		11.79	13.00
Std. Dev.	2.27	2.36		3.74	4.03		2.43	1.83		2.99	3.02

Table 2: Problem Solving Comparison Between Problem Solving and RAPTOR Sections.

The above data shows that students in the RAPTOR programming section scored consistently higher in both their raw and problem solving scores than students in sections where only problem solving techniques were taught. Table 3 shows the comparison of the p-values obtained when comparing the overall assignment and problem solving scores between the sections.

	Raw Score	Problem Solving Score
P-value	0.003	0.006

Table 3: P-values for Overall Problem Solving and Raw Scores.

In addition, student interest in computing was measured through the surveys given at the beginning and end of the semester. Students were asked how likely they were to take more computing courses in the future. Analysis of these surveys shows that, overall, student interest in taking another computing course declined slightly after taking CIT 110. However, interest declined more in the sections where problem solving was taught, compared to the section taught with RAPTOR. Table 4 shows that while the percentage of students who lost interest in computing was about the same regardless of the teaching methodology, the percentage of students who became more interested in computing was nearly double in the section where RAPTOR was taught.

	RAPTOR Section	Problem Solving Sections
Percent of Students with Increased Interest	17.39%	9.30%
Percent of Students with Decreased Interest	30.43%	30.23%

Table 4: Change in Student Interest in Computing.

We have been working on identifying how successful students from each section of CIT 110 have been in our CS1 course. However, only three students from the spring have completed the CS1 course at this point, so there is insufficient data available.

2.3 Analysis

The results in Table 3 indicate that there is a statistically significant difference in both the raw and problem solving scores, favoring students who were taught RAPTOR programming. It would be expected that giving students additional experience in using a program like RAPTOR would translate to success in other applications. However, the success of the RAPTOR students in problem solving is especially interesting given that one might expect that the techniques taught explicitly in the problem solving unit would have translated to better problem solving techniques on the assignments. It is possible that the students who used RAPTOR developed an increased awareness of the need for well-defined parameters in a RAPTOR program when making the function calls necessary for their graphics and this translated to an understanding of the importance to include this in the solutions they were developing in Microsoft Office applications. In addition, this result could be due to students' level of comfort with the idea of calling functions. This could then translate into a less severe learning curve when transitioning to Office.

In addition, the larger number of students who increased their interest in computing is likely due to the opportunity to easily create graphics and animations in RAPTOR. The fun of computing is much more apparent in RAPTOR programming than it is in working through problem statements and creating flowcharts, even though that is a valuable skill.

3 Further Work in CIT 110

In addition to the use of RAPTOR in CIT 110, we have been investigating other techniques to enhance student learning. During the fall of 2011, sections of CIT 110 were again taught beginning with either RAPTOR programming or problem solving. There were two common assignments that were graded using a common grading rubric, but not a problem solving rubric. Overall, assignment scores in the problem solving sections were higher than those in the RAPTOR section, but the difference was not statistically significant.

3.1 Electronic Textbooks

Also during the fall of 2011, we taught CIT 110 without a traditional textbook and instead utilized MyITLab through Pearson Higher Education [7]. This was chosen because MyITLab provided the flexibility to choose specific electronic chapters from many of their textbooks. This included texts involving computing concepts as well as Microsoft Office applications, which fit the broad range of topics covered in CIT 110 quite well. MyITLab also provided many online training simulations in Microsoft Excel and Access, as well as online learning materials for computing concepts in networks and hardware. Overall, the instructors appreciated the flexibility that MyITLab provided in being able to choose chapters from many different books. However, based on comments on surveys and classroom discussions, many students would have preferred having a traditional textbook instead of an electronic one. In the end, we decided to move to a custom textbook through Pearson, which provided the desired flexibility, but also provided students with a more traditional text.

In addition, while the online simulation exercises in MyITLab were valuable, students were not able to take as much advantage of them as we had initially hoped. While students were able to successfully complete the online exercises, they were not able to apply the concepts from the simulations to new situations. This is probably because they only focused on completing the next step in the simulations and failed to see the big picture. In addition, the more complicated, automatically graded assignments provided through MyITLab were not as flexible in grading as we needed them to be, and we were not able to utilize them. We have now returned to a more traditional method of teaching Microsoft Office applications.

3.2 Future Work

Since RAPTOR programming appears to be an improvement over the problem solving technique, we will continue to work to integrate it into our curriculum for CIT 110. Although at this time it still does not seem practical to incorporate RAPTOR into all the sections of the course, given different instructor preferences and comfort with teaching RAPTOR. Fortunately, this will provide the opportunity to continue to compare the effectiveness of teaching RAPTOR programming instead of problem solving. In addition, we will continue to follow up on students who are continuing on to CS1 to see whether there is any correlation between the teaching method of CIT 110 and success in CS1.

There are some other issues that would need to be addressed in order to fully integrate RATOR programming into CIT 110. First, the two weeks available to teach RAPTOR does not seem sufficient to give students an opportunity to develop much depth in learning RAPTOR. Before the changes in 2010, instructors who had introduced RAPTOR did so by spending three or four weeks of class covering it. Therefore, two weeks is barely sufficient to introduce students to RAPTOR and allow them to create in-depth programs. Also, many of the topics taught in the problem solving component have value beyond the realm of computing, and those themes have a broader benefit for those students who are not majors in Computer Science or Management Information Systems. In the future, we would like to look into ways to integrate both the programming and problem solving components, if we can do it in such a way to maintain their benefits and also continue to effectively cover the other outcomes of the course.

References

[1] J. Barr, S. Cooper, M. Goldweber and H. Walker, "What Everyone Needs to Know About Computation," in *SIGCSE 2010*, 2010.

[2] S. Davies, J. A. Polack-Wahl and K. Anewalt, "A Snapshot of Current Practices in Teaching the Introductory Programming Sequence," in *SIGCSE 2011*, 2011.

[3] M. C. Carlisle, "RAPTOR: a visual programming environment for teaching algorithmic problem solving," in *SIGCSE 2005*, 2005.

[4] S. Thompson, "Where do I begin? A Problem Solving Approach in Teaching

Functional Programming," in *PLIP '97*, 1997.

[5] G. Polya, How To Solve It, Second ed., Princeton, New Jersey: Princeton University Press, 1957.

[6] Schreyer Institute for Teaching Excellence, "Problem Solving Rubric," 2007. [Online]. Available: http://www.schreyerinstitute.psu.edu/pdf/ProblemSolvingRubric1.pdf. [Accessed January 2011].

[7] Pearson Higher Education, "MyITLab," [Online]. Available: http://www.myitlab.com. [Accessed May 2011].