Adrienne Decker
Department of Computer Science & Engineering
University at Buffalo
adrienne@cse.buffalo.edu

How Students Measure Up: Creation of an Assessment Instrument for Introductory Computer Science

# Introduction

As computers and computing began to emerge as a field in the middle of the last century, colleges and universities began creating departments and degree programs in this field of study. As the field has evolved, it has been directed by three curriculum documents (ACM/IEEE-CS Joint Curriculum Task Force Curricula, 1991; Committee on Computer Science Curriculum, 1968, 1978).

The most recent of these has been Computing Curricula 2001, more commonly known as CC2001 (Joint Task Force on Computing Curricula, 2001) . Before CC2001, there was much debate in the literature about the approach, assignments, lab environments and other teaching aides that were most appropriate for courses. Of special interest was the introductory sequence of courses (CS1-CS2), due to the fact that these were the first courses that students were exposed to. CC2001 legitimizes six approaches to the introductory sequence, which include three programming-first approaches: Imperative-first, Objects-first, and Functional-first as well as Breadth-first, Algorithms-first, and Hardware-first. The report does not recommend one over the other, but rather points out the relative strengths and weaknesses of all of them.

CC2001, as with the previous curricula, does not provide faculty with instructions for how to implement the suggestions and the guidelines contained within. This leaves faculty to take their own approaches to the material, and invent assignments, lab exercises and other teaching aides for specific courses outlined in the curriculum. Whenever a new curricular device is conceived, the next natural step in the investigation is to see if the innovation actually helps student's understanding of the material. Investigations into some of these innovations has previously been measured by lab grade, overall course grade, resignation rate or exam grades (Cooper, Dann, & Pausch, 2003; Decker, 2003; Ventura, 2003)

The problem with using these types of metrics in a study is that often they are not proven reliable or valid. Reliability, or the degree of consistency among test scores, and validity, the relevance of the metric for the particular skill it is trying to assess are both essential whenever the results of these metrics are to be analyzed (Kaplan & Saccuzzo, 2001; Marshall & Hales, 1972; Ravid, 1994) . Also, within these types of studies, it is not often specified how a particular grade is arrived at. For example, when using overall course grade as the success marker, one should know if there was a curve placed on the grades, or even the basic breakdown of what is considered "A" work.

With all of the claims of innovation in CS1 curriculum, we need a way of assessing student's comprehension of the core CS1 material. The goal of this work is to create a reliable and validated assessment instrument for CS1. The test will be one that assesses the knowledge of a student who has taken a CS1 class using one of the programming-first approaches described in CC2001. This assessment should be independent of both the approach used for CS1 and should not rely on testing a student's syntactic ability with a particular language.

# Theoretical Background & Previous Research in the Area

Before CC2001, there was a long debate over what is the most acceptable way to teach the introductory computer science curriculum. Many have contributed to the myriad of approaches for teaching the introductory sequence of courses. Owens, Cupper, Hirshfield, Potter, and Salter (1994) offered varying viewpoints on the different models for teaching CS1. Evans (1996) also offers a model for the CS1 curriculum that emphasizes using topics that pervade the entirety of the computer science domain. The approach given by Proulx, Rasala and Fell (1996) is similar to the one given by Alphonce and Ventura (2003). Both groups advocate an approach to CS1 that utilizes graphics and graphical programming to motivate the core material in CS1. Each approach has merit and gives anecdotal evidence for their success, but it is necessary to have adequate ways to assess student knowledge of the concepts of introductory programming to bolster arguments for the success of the approach.

The need for accurate assessment once again reveals itself when one looks at the literature on predictors of success for CS1 (Evans & Simkin, 1989; Hagan & Markham, 2000; Kurtz, 1980; Leeper & Silver, 1982; Mazlack, 1980; Wilson & Shrock, 2001) . For each of these studies, different factors were identified as possible reasons for success in a programming-first CS1 course. In each case, success was measured either by overall exam score, laboratory exercise scores, programming assignment scores, or exam scores. None of these measures of success were validated, or clearly explained.

Even in recent work done on a course that embraces CC2001's recommendations for an Objects-first CS1 only uses measures of overall course grade, exam grades, and lab grades in its study (Ventura, 2003) . The predictive values of the factors studied are given as in the other work cited above, but it once again fails to convince that the level of success in the students has been validated in some form.

There has been one documented attempt at creation of an assessment for CS1. The working group from the Conference on Innovation and Technology in Computer Science Education (ITiCSE) in 2001, created a programming test that was administered to students at multiple institutions in multiple countries (McCracken et al., 2001) . The group's results indicated that students coming out of CS1 did not have the skills that the test assessed.

One of the positives about this attempt at assessment is that it included problems that were well thought out and made an attempt to cover all of the material that a CS1 student should have mastery of. Another positive was the fact that there were specific grading rubrics created for the problems that helped lead to uniform scoring. The students were not restricted to a particular language or programming environment, so the students completed the exercises in whichever way was most comfortable to them.

However, the study was flawed as recognized even by the participants. The problems given had an inherent mathematical flavor that would have disadvantages students with mathematical anxiety. They also admit in their analysis that one of the test questions "was undoubtly difficult for students who had never studied stacks or other basic data structures." (McCracken et al., 2001) They also pointed out flaws in the presentation of the problems and the instructions for administering the exercises. Therefore, even with all the positives of this study, there is still room to grow and make an assessment instrument that could be more true to the current flavors of CS1 as described in CC2001.

# Goals of the Research

The ultimate goal of the research is to create a validated and reliable metric for assessing student's level of knowledge at the completion of a programming first CS1. The test should be language and paradigm independent. This test will then be available to assess not only student progress, but also as a way to gauge particular pedagogical advances and their true value within the classroom.

The current hypotheses are:

- The current methods for testing pedagogical innovations are not adequate.
- An intersection of topics can be identified for the three programming-first approaches to the introductory curriculum
- A test can be written to assess a student's level of achievement with the introductory curriculum.

# Current Status & Stage in Program

My proposal has been approved and I am now working on the actual pieces of creating the assessment instrument. Hopefully, the instrument will be ready for its first administration at the end of the Spring 2005 semester.

# Interim Conclusions

The analysis of CC2001 has been completed and an intersection of adequate size has not been found for the topics in CS1 in the programming-first approaches. Therefore, the search for the intersection was expanded to include CS1 and CS2. Also, learning objectives have been identified for each of the topics included in the intersection. After completing this list, it was realized that there were simply too many topics for one

instrument. Also, many of the topics and learning objectives sought in the introductory curriculum do not lend themselves to assessment in a traditional test manner. We are now in the process of refining the list of topics to one that is more manageable for the creation of the test.

# Open Issues

Open issues for this research include

- Can there be an assessment tool that accurately assesses student's progress through a curriculum, given the differences across schools and curriculum?
- Can the non-programming first approaches be assessed using the same metric at the end of the introductory curriculum?
- Can information gathered using this assessment tool help us determine if a particular pedagogical innovation is helping the students learn?

# What I Hope to Gain From Participation in Doctoral Consortium

I hope to gain input and feedback about my research ideas. I am also hoping for informed guidance on the approach I am taking towards my research and suggestions on how to proceed forward.

# Bibliographic References

ACM/IEEE-CS Joint Curriculum Task Force Curricula. (1991). *Computing curricula 1991*. IEEE Computer Society & Association for Computing Machinery. Retrieved October 30, 2003, from the World Wide Web: http://www.computer.org/education/cc1991

Alphonce, C. G., & Ventura, P. R. (2003). *Using Graphics to Support the Teaching of Fundamental Object Oriented Principles.* Paper presented at the OOPSLA 2003 Educator's Symposium, Anaheim, California.

Committee on Computer Science Curriculum. (1968). Curriculum 68: Recommendations for the undergraduate program in computer science. *Communications of the ACM, 11*(3), 151-197.

Committee on Computer Science Curriculum. (1978). Curriculum 78: Recommendations for the undergraduate program in computer science. *Communications of the ACM, 22*(3), 147 - 166.

Cooper, S., Dann, W., & Pausch, R. (2003). *Teaching objects-first in introductory computer science.* Paper presented at the 34th SIGCSE technical symposium on Computer Science Education, Reno, Nevada.

Decker, A. (2003). A tale of two paradigms. *Journal of Computing Sciences in Colleges, 19*(2), 238-246.

Evans, G. E., & Simkin, M. G. (1989). What best predicts computer proficiency? *Communications of the ACM, 32*(11), 1322 - 1327.

Evans, M. D. (1996). A new emphasis & pedagogy for a CS1 course. *SIGCSE Bulletin, 28*(3), 12 - 16.

Hagan, D., & Markham, S. (2000). *Does it help to have some programming experience before beginning a computing degree program?* Paper presented at the 5th annual SIGCSE/SIGCUE conference on Innovation and technology in computer science education.

Joint Task Force on Computing Curricula. (2001). *Computing curricula 2001 computer science*. IEEE Computer Society & Association for Computing Machinery. Retrieved October 30, 2003, from the World Wide Web: http://www.computer.org/education/cc2001/final/index.htm

Kaplan, R. M., & Saccuzzo, D. P. (2001). *Psychological Testing: Principles, Applications and Issues* (Fifth ed.). Belmont, California: Wadsworth/Thomson Learning.

Kurtz, B. L. (1980). *Inivestigating the relationship between the development of abstract reasoning and performance in an introductory programming class.* Paper presented at the 11th SIGCSE technical symposium on Computer Science Education, Kansas City, Missouri.

Leeper, R. R., & Silver, J. L. (1982). *Predicting success in a first programming course.* Paper presented at the 13th SIGCSE technical symposium on computer science education, Indianapolis, Indiana.

Marshall, J. C., & Hales, L. W. (1972). *Essentials of Testing*. Reading, Massachusetts: Addison-Wesley Publishing Co.

Mazlack, L. J. (1980). Identifying potential to acquire programming skill. *Communications of the ACM, 23*(1), 14 - 17.

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., Laxer, C., Thomas, L., Utting, I., & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of the first-year CS students. *SIGCSE Bulletin, 33*(4), 1 - 16.

Owens, B. B., Cupper, R. D., Hirshfield, S., Potter, W., & Salter, R. (1994). *New models for the CS1 course: What are they and are they leading to the same place?* Paper presented at the 25th SIGCSE symposium on Computer science education, Phoenix, Arizona.

Proulx, V. K., Rasala, R., & Fell, H. (1996). *Foundations of computer science: What are they and how do we teach them?* Paper presented at the 1st conference on Integrating technology into computer science education, Barcelona, Spain.

Ravid, R. (1994). *Practical Statistics for Educators*. Lanham: University Press of America.

Ventura, P. R. (2003). *On the origins of programmers: Identifying predictors of success for an objects-first CS1*. Unpublished Doctoral, University at Buffalo, SUNY, Buffalo.

Wilson, B. C., & Shrock, S. (2001). *Contributing to success in an introductory computer science course: A study of twelve factors.* Paper presented at the 32nd SIGCSE technical symposium on Computer Science Education, Charlotte, North Carolina.