Effectiveness of a Computational Thinking (CS0) Course on Student Analytical Skills

Michele Van Dyne
Montana Tech of the University of Montana
1300 W. Park St.
Butte, MT 59701
1-406-496-4855
mvandyne@mtech.edu

Jeffrey Braun
Montana Tech of the University of Montana
1300 W. Park St.
Butte, MT 59701
1-406-496-4206
jbraun@mtech.edu

ABSTRACT
In this paper, we describe the content and evaluation of a Computational Thinking (CS0) course developed to improve the analytical problem solving of students participating in the course. The course is targeted to students who are mathematically under-prepared to enter our introductory programming sequence; however, it has recently been included in the University's general education curriculum so that students majoring in any discipline may take the course. Using the Whimbey Analytical Skills Inventory (WASI) students in the CS0 class, along with students in an analogous level engineering class (FESP), were tested at the beginning of the course and again at the end, using different versions of the test. The improvement in scores was statistically significant when measured by both the student t-test and the Cohen d (effect size) for CS0 students but not for the FESP students, providing support that the course does, in fact, increase student analytical problem solving skills. Courses in Computational Thinking have demonstrated success in many schools; however, this research demonstrates its effectiveness in improving analytical skills in majors as well as non-majors.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education

General Terms
Measurement, Performance, Experimentation

Keywords
Computational thinking, problem solving, analytical skills, critical thinking, course effectiveness, CS0

1. INTRODUCTION
The introductory computer science sequence (CS1 and CS2) at Montana Tech does not require any previous programming experience, but it does have a corequisite of at least pre-calculus level math (ACT Math score of 24 – 26). We find that students weak in math skills typically struggle in our CS1 and CS2 courses. To prepare these students for computer science studies, we developed a CS0 level Computational Thinking course. This course primarily covers problem solving and critical thinking skills to assist students in the analytical skills they will need to complete CS1. In initial offerings of our CS0 course, students were those who majored in either computer science or software engineering, but because of its demonstrated success in enhancing analytical and problem solving skills, the University has approved the course as a general education course. Now students majoring in other areas also take the course.

There is a similar course offered to mathematically underprepared students entering traditional engineering programs. This course, FESP 095 (FESP stands for Foundations of Engineering and Science Program), is designed to teach students the skills they will need to complete engineering and science programs.

The computer science education literature shows many different approaches to CS0 courses for both majors and non-majors [2, 3, 7, 10, 13]. Some authors (e.g., Middleton [9]) simply describe their introductory course design and the practical issues in presenting it in order to share their novel teaching techniques. Many researchers try to assess the effectiveness of their CS0 course by examining CS0 student grades and/or retention in CS1 and CS2 courses. Dierbach [3] shows that the group of students taking Towson University's General Computer Science (CS0) course outperformed all other student subgroups in CS1. Rizvi [13] showed Scratch programming based CS0 students performed well in CS1. Huangs [7] assessed Cal Poly's new CS0 course by examining grades in CS1/CS2 and comparing the CS2 retention rates of students who did not take CS0 to those that did.

Other CS0 courses [2, 9, 10] similar to Computational Thinking focus less on computer programming and more on problem solving and critical thinking. Researchers have long recognized the importance of incorporating problem solving into introductory computer science courses [8]. Middleton [9] and Mitchel [10] both describe Problem Solving courses at their institutions and monitor their CS0 students in subsequent courses. Middleton's course uses robots "as the environment for the problem solving because they make errors obvious, better motivating students to finish activities completely". Middleton also concludes that robots are more fun and may improve retention. The Principles of Computation course at Carnegie Mellon "focuses on the principle of computation, and how computer scientists study computation

SIGCSE '14, March 05 - 08 2014, Atlanta, GA, USA

978-1-4503-2605-6/14/03…$15.00.
http://dx.doi.org/10.1145/2538862.2538956

through the design and analysis of algorithms, correctness and efficiency, the limits of computation, and several unique applications that build on CS ideas (cryptography, artificial intelligence)" [2]. The assessment for this CS0 course relies on grades and student feedback as mostly non-majors enroll in it and do not move onto CS1.

Other disciplines (e.g., writing, psychology, philosophy) have assessed the effectiveness of their courses by using standardized tests to measure changes in critical thinking and analytical skills. Numerous standardized critical thinking tests exist to assess students [5, 11, 12]. Possin [12] reviews the different critical thinking assessment methods and many of the popular standardized tests, including simple to administer multiple-choice tests. He concludes that objective multiple-choice tests appear to be "fairly accurate for measuring students" acquisition of specific core CT skills".

To measure the effectiveness of a course for improving critical thinking skills, researchers typically give a pre- and post-course test. In the critical thinking literature, the popular way to report the practical significance of the difference from one test to the next within a group is to compute the ratio of the difference in mean scores to the standard deviation [4]. This ratio is often called the *effect size* or Cohen's d. Critical thinking effect sizes for a small subset of writing and psychology courses range from 0.24 of a standard deviation (or 0.24 SD) to 1.47 SD [6, 12, 14]. Hatcher [6] compares three different critical thinking tests and shows the effect sizes for groups varies from 0.57 SD to 0.97 SD depending on which test is administered. Solon [14] infused general critical thinking material into an introductory psychology course and showed significant change ($p=0.000004$) and an effect size (Cohen's d) = 0.87 versus the control group which was not significant ($p=0.49$) and effect size (Cohen's d) = 0.10.

2. COURSE CONTENT

2.1 Overview of the Course

Following the lead of Jeannette Wing of Carnegie Mellon University (CMU) [15], we introduced a course called Computational Thinking designed to teach problem solving skills to underprepared students entering the curriculum. Diverging from the CMU approach, but consistent with Middleton's [9] approach, we also incorporated simple robotics in the course offering that includes the lab. Students may take the course as a 2 credit lecture class or a 3 credit lecture and lab class. The course is taught as two hours of lecture and three hours of lab per week. The catalog description is:

> "Computational thinking involves solving problems, designing systems, and to understanding human behavior, by drawing on the concepts fundamental to computer science. It is the study of an effective approach used by people to solve problems. Critical thinking involves the systematic evaluation of information, and is a crucial piece of problem solving. The two are combined in this course to provide the student with a powerful set of tools to understand and solve the kinds of problems they will encounter in their college studies and future careers. The lab incorporates a programming component In this programming lab, students learn to carefully and systematically analyze problems and demonstrate the correctness of their solution by implementing it in program code."

The Computational Thinking course has been through four full offerings, and is in its fifth offering at present. Last year's offering is the first time that the course has been included in Montana Tech's general education curriculum, and its enrollment saw a significant increase.

2.2 Lecture Content

The lecture portion of the course stresses logical and mathematical thinking and problem solving. The topics covered during lecture and reinforced through weekly class assignments include analogies, analysis of trends and patterns, logical reasoning, mathematical word problems, and critical evaluation of information in addition to computer science oriented topics such as algorithm development, data type and structure, abstraction, decomposition, transformations, simulation, iteration, recursion and the introduction of specific algorithms.

The in-class portion of the course is taught very interactively. During each class, students are given about 15 minutes of instruction on a topic then the class as a whole participates in exercises about that topic. An attempt is made to get all students to actively participate. The class is then broken into groups which work on additional exercises. Weekly assignments are given which also reinforce the concepts taught that week.

Material for the course has been drawn from a variety of sources, though much of the computer science oriented topics have been drawn from Backhouse's *Algorithmic Problem Solving* [1], and from pre-publication material from this source available online.

2.3 Lab Content

The lab portion of the CS0 course uses the Lego Mindstorms NXT robotics kits to illustrate concepts taught during lecture. The robotics approach was chosen rather than general programming because of the interactive and visual feedback of results to students. Students in the lab must build their own robots and program them, first using the graphical programming environment provided with the kits, and then moving on to programming them in Java (using LeJos, a Java library for the NXT robots). Lab assignment topics to reinforce the lecture topics include programming as reasoning, programming as math, algorithms, data and variables, decisions, problem solving, functions, iteration, sorting and a final assignment of tying it all together.

As one example of a lab assignment, students are asked to program their robot, which at this point has distance sensing capability, as a "feral robot". The robot is to continually sense objects within a certain distance. If something enters this "unsafe" distance, the robot is to back away, unless the intruder gets even closer, at which point the robot is to attack. The definition of attack behavior is left to the student. This assignment incorporates looping and decision constructs along with the development of the algorithmic logic. The first version of this assignment is done using the graphical programming environment, and a later version is rewritten using Java code.

Students are not required to take the lab, but most do. In the past four offerings, 74% of the students took both the lecture and lab portion of the course.

3. EVALUATION

To test the effectiveness of incorporating the Computational Thinking course into the pre-curriculum, the Whimbey Analytical Skills Inventory (WASI) pre-test and post-test [16] were used in each of the four complete offerings of the course. On the first day of class, the WASI pre-test was administered. It consists of 38 problems that test mathematical and logical thinking in various forms. During the semester, instruction is given which first addresses the types of problems the WASI test covers, and then delves into deeper logical concepts such as algorithm development, using structured logic, invariants, and recursion. Students are not informed they will be taking either the WASI pre-test or post-test; in fact, they are told that the final exam will cover class material, and in fact, the final exam is completely separate from the WASI exams. Furthermore, the WASI tests have no bearing on student grades, and students are informed of this when they take the test. On the last day of classes, the post-WASI was administered. This is a second version of the WASI test, consisting of 37 problems of similar composition, but different content, as the pre-test.

It is possible that any improvement in student performance in the CS0 class could be attributed to some factor outside of the class, such as completion of the first semester of college coursework. To test for this external effect, we also administered the pre- and post-tests to students in the pre-engineering course, FESP 095. Students in the FESP program have a similar background to those that take the CS0 course, in that they are mathematically underprepared to enter into the introductory engineering classes. The FESP 095 course also has similar goals to CS0, that is, to prepare students with the problem solving skills they will need to complete an engineering or science degree. The catalog description of this course is:

> "This course will focus on the skills that are necessary to successfully enter and complete an engineering or science program at Montana Tech. These skills include dimensional analysis, unit conversion, technical writing, and technical drawing. Additional components included in the curriculum are presentation of technical information and applications/problem solving of Intermediate Algebra."

The WASI tests contain questions that are categorized into several categories: verbal reasoning, sequential instructions, analogies, relationship sentences, analysis of trends and patterns, and mathematical word problems. None of these problems requires mathematical skills beyond basic algebra, though most problems require logical or analytical skills and attention to detail.

WASI pre-test and post-tests were included for analysis only if a student took both of the tests. (Not all students are present for the first day of classes, nor do all students attend class, particularly if they are unaware that an exam will be administered.) From the CS0 course, an additional set of pre-test/post-test scores was discarded because one student misunderstood the instructions and answered none of the questions on the pre-test, resulting in a 33 point gain between pre- and post-test scores. Including this data point would have skewed the data favorably, but would have been a misrepresentative sample. Students who enrolled in both the CS0 course and the FESP 095 course and completed the test in both were not included in the data analysis.

It is possible that students who completed both the pre-test and post-test in either the CS0 or the FESP 095 course were more motivated than those who did not complete both tests; that is, these are the students that regularly attended class. By including the same group from each course, we are at least able to measure the effect on those motivated students across different class environments.

4. RESULTS AND DISCUSSION

Over the four completed offerings of the CS0 course, 35 total students enrolled; 22 of these students completed both the pre- and post-tests. In the FESP 095 course, 52 students enrolled; 29 of these students completed both versions of the test. To provide incentive for the FESP 095 students to take the post-test seriously, they were given extra credit if their post-test scores improved. A similar incentive was not offered to CS0 students. There were no students who were enrolled in both classes that completed both tests in both classes.

A student t-test was used to compare pre-test scores in the FESP 095 course and those from the CS0 course to test whether we were testing similar populations. There was a slight difference, but the difference was not significant ($p=0.056$), giving us evidence that we are comparing a similar population of students.

In the FESP 095 course, initial WASI pre-test raw scores ranged from a low of 8 to a high of 31 out of 38 possible, with an average score of 20, while post-test scores ranged from a low of 12 to a high of 30 out of 37 possible, with an average score of 21. Translating this to percentages, the pre-test had a range of 21.1% to 81.6% , and the post-test had a range of 32.4% to 81.1%. The average point improvement over the course of a semester was .93, and the average percentage point improvement was 4%. Distribution of the scores for the pre- and post-tests can be seen in Figures 1 and 2 below.

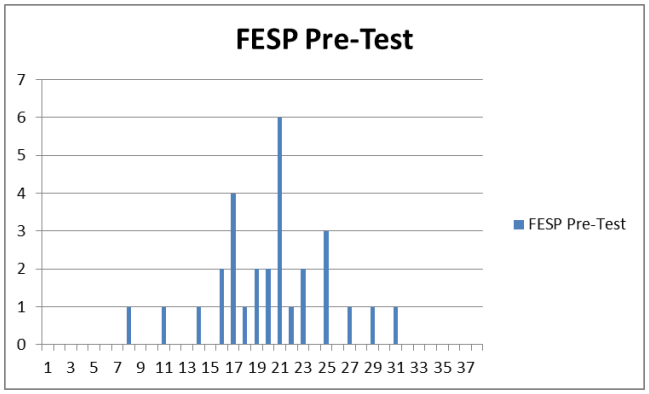

Figure 1. FESP Pre-Test Score Distribution

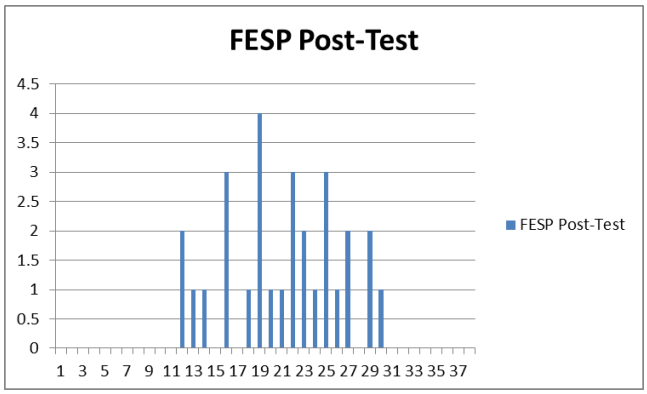

Figure 2. FESP Post-Test Score Distribution

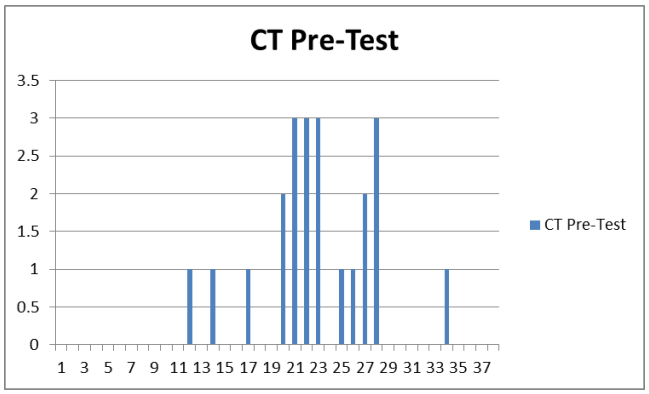

Figure 3. CS0 Pre-Test Score Distribution

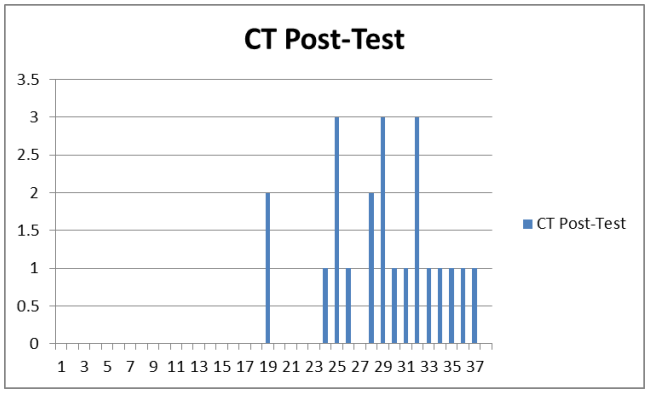

Figure 4. CS0 Post-Test Score Distribution

In the CS0 course, initial WASI pre-test raw scores ranged from a low of 12 to a high of 34 out of 38 possible, with an average score of 23, while post-test scores ranged from a low of 19 to a high of 37 out of 37 possible, with an average score of 29. Translating this to percentages, the pre-test had a range of 31.6% to 89.5%, with an average of 60.3%, while the post-test had a range of 51.4% to 100%, with an average of 78.4%. The average point improvement over the course of a semester was 6.09, and the average percentage point improvement was 18.1%. Distribution of the scores for the pre- and post-tests can be seen in Figures 3 and 4 below.

To compare differences between pre- and post-test scores, the student's t-test was used. The percentage correct on both WASI test was used in order to adjust for any differences caused by the number of questions on each test, and thus a difference in raw scores.

In the FESP 095 course, a test of significance between scores using a two-tailed, paired test, since we are not sure of which direction a change would occur, reveals no significant difference between pre- and post-test scores ($p=0.098$). If we assume that change should occur in one direction (positive) and not the other, and using a single-tailed paired test, we get marginal significance ($p=0.05$).

For the CS0 course, assuming the change between test scores should be in the positive direction, and thus using a single-tailed, paired test, a significant difference was found ($p=0.00000014$). If we make the assumption that we don't know the direction of the change, and test for significance using a two-tailed paired test, we still get a significant difference ($p=0.00000028$). This provides evidence that participation in the course had a significant effect in improving analytical skills as tested by the WASI.

As discussed previously in this paper, in the critical thinking literature, a common way to report the practical significance of the difference from one test to the next within a group is to compute the ratio of the difference in mean scores to the standard deviation [3], called the *effect size*. For our data, the effect size was calculated to be 1.36 SD for the CS0 course and 0.29 SD for the FESP 095 course. Reported effect sizes on research in the literature ranged from 0.24 SD to 1.47 SD, with researchers expressing satisfaction at effect sizes over 0.5 SD. Our effect size of 1.36 SD lends additional support that participation in the course increased student abilities in analytical skills as measured by the WASI. The FESP 095 control group's effect size of 0.29 SD is within the range (0.23 SD to 0.44 SD) reported for several Critical Thinking courses summarized by Possin[12].

It was previously mentioned that not all students take both the lecture and lab portion of the class; approximately 26% of students take the lecture portion only. Among the students in this study, then, 6 took the lecture portion only, and 16 took both lab and lecture. Although these numbers are getting smaller, we looked at the differences between these two groups. There was no difference at all in pre-test performance between the two groups ($p=0.478$). There is a significant, $p<0.05$, though small difference between the groups, however, in post-test performance ($p=0.037$). That is, those students who also attended lab performed better than those who attended lecture only. Finally, the difference between pre- and post-test performance for those students who only attended lecture was significant ($p=0.0008$), and an even more significant difference was found in the test score of those

who attended both lab and lecture (p=0.00004). The effect sizes for these two subgroups are 1.19 SD and 2.97 SD respectively.

5. CONCLUSIONS AND FUTURE WORK

Statistical analysis of our results give promising indications that the CS0, Computational Thinking, course does, in fact, positively and significantly impact the analytical problem solving skills of students that participate. And although the course was initially developed to assist underprepared students who are majoring in computer science and software engineering, particularly since it has been opened as a general education course, we have other majors enrolling in the course. In the data used in this study, 10 of the CS0 students were those declaring a major from our department, while other majors included Health Care Informatics, Electrical Engineering, Chemistry, General Engineering and Petroleum Engineering. This suggests that the result of teaching Computational Thinking improves the analytical ability of students in general, and not just those predisposed to being computer science majors. As a subjective side note, most comments on course evaluations have been very positive. One recent student remark was "I have learned new concepts of problem solving that I feel will be crucial to me later on in life."

We plan on continuing to collect pre- and post-test data on students taking the CS0 class, particularly as our enrollment grows and even more non-majors take the course. We think this course can be an effective recruiting tool in that it introduces all students to the types of problem solving required in computer science. On the other hand, since the impact on increased analytical skills which are not computationally specific has been demonstrated, we would like to see more students encouraged to take a course of this nature to help them in problem-solving in general.

6. REFERENCES

[1] Backhouse, R. 2011, *Algorithmic Problem Solving*, John Wiley & Sons, West Sussex, UK, 2011.

[2] Cortina, T. 2007, An introduction to computer science for non-majors using principles of computation, *ACM SIGCSE Bulletin* 39, 1 (March 2007), 218-222.

[3] Dierbach, C., Taylor, B., Zhou, H. and Zimand, I. 2005, Experiences with a CS0 course targeted for CS1 success, *ACM SIGCSE Bulletin* 37, 1 (March 2005), 317-320.

[4] Ennis, R. H. 2008. Nationwide Testing of Critical Thinking for Higher Education: Vigilance Required, *Teaching Philosophy* 31, 1 (March 2008), 1-26.

[5] Follman, J., Lavely, C., and Berger, N. 1997, Inventory of instruments of critical thinking, *Informal Logic* 18, 2, 261-267.

[6] Hatcher, D. L., 2011, Which test? Whose scores? Comparing standardized critical thinking tests, *New Directions for Institutional Research* 2011, 149, 29-39.

[7] Haungs, M., Clark, C., Clements, J., and Janzen, D. 2012. Improving first-year success and retention through interest-based CS0 courses, *SIGCSE '12 Proceedings of the 43rd ACM technical symposium on Computer Science Education* (Feb. 2012), 589-594.

[8] Hyde, D. C., Gay, B., and Utter, D. 1979. The integration of a Problem Solving Process in the first course, *SIGCSE '79: Proceedings of the tenth SIGCSE technical symposium on Computer science education* (Feb. 1979), 54-59.

[9] Middleton, D. 2012. Trying to teach problem-solving instead of just assigning it: Some Practical Issues. *Journal of Computing Sciences in Colleges* 27, 5 (May, 2012), 60-65.

[10] Mitchell, W. 2001. Another look at CS0, *Journal of Computing Sciences in Colleges* 17, 1 (Oct. 2001), 194-205.

[11] Morante, E. A., Ulesky, A. 1984. Assessment of Reasoning Abilities, *Educational Leadership* 42, 1 (Sept. 1984), 71-74.

[12] Possin, K. 2008. A Field Guide to Critical-Thinking Assessment, *Teaching Philosophy* 31, 3 (Sept. 2008), 201-228.

[13] Rizvi, M., Humpries, T., Major, D., Lauzun, H., and Jones, M. 2011. A new CS0 course for at-risk majors, *24th IEEE-CS Conference on Software Engineering Education and Training* (May 2011), 314-323.

[14] Solon, T. 2007. Generic critical thinking infusion and course content learning in introductory psychology, *Journal of Instructional Psychology* 34, 2 (June 2007), 95-109.

[15] Wing, J. M. 2008. Computational thinking and thinking about computing," *Philosophical Transactions of the Royal Society*, 366 (July 2008) 3717-3725.

[16] Whimbey, A. and Lochhead, J. 1999. *Problem Solving and Comprehension*, 6th ed., Psychology Press, Taylor and Francis Group, New York and London, 1999.