2017

Predictors of Success in Learning Computer Programming

Deborah K. Mathews
University of Rhode Island, deborah_mathews@my.uri.edu

PREDICTORS OF SUCCESS IN LEARNING COMPUTER PROGRAMMING

BY

DEBORAH K. MATHEWS

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE

UNIVERSITY OF RHODE ISLAND

2017

DOCTOR OF PHILOSOPHY DISSERTATION

OF

DEBORAH K. MATHEWS

APPROVED:

Dissertation Committee:

Major Professor Edmund Lamagna

Gerard Baudet

Minsuk Shim

Nasser H. Zawia
DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND
2017

Abstract

Worldwide, beginning programming has a success rate of 67.7%, which may be a barrier to success for aspiring computer science majors. This may be particularly problematic for women, since only 18% of U.S. graduates with a bachelor degree in computer science are women. The purpose of this study is to identify conceptual predictors of success so those most likely to struggle can be identified. Specifically the focus is on the relationship of math to learning programming. Are math prerequisites helpful or a barrier to success? Can math achievement be used as a predictor of success, and can it be used to detect discrepancies between the success of men and women learning to program? The method for determining the best measure of math achievement is correlational comparison. The method for determining the effect of learning math is comparison of means. The method for determining predictors of success is linear regression.

The results are that the best measure of math achievement is the average of math grades from the year prior to the programming course. Either this average or the grade in the most recent math course can be used to predict whether a student is likely to achieve above a threshold of success. The math average can be used to detect discrepancies between the performance of men and women. For those who have taken a math course in the prior year, the predictor that is most significant is GPA at the time of taking the programming course. For those who start out in precalculus, those who do poorly do not improve their performance by retaking precalculus and doing better. Nor do students who take more math beyond precalculus see significant improvement.

But for those who start out in calculus, those who struggle do see improvement when they succeed in learning calculus, and they do benefit from learning more math beyond calculus.

Acknowledgments

I would like to thank my advisor, Professor Edmund Lamagna, for acting as a guide on the path starting from never having done research to completing the Masters thesis and now this dissertation. He has notified me of opportunities and encouraged me to take them. His critical analysis has been indispensable. He has helped me to see the bright side when the results were not what I hoped for, see next steps and explain unexpected results.

I would also like to thank my mentor, Professor Gérard Baudet, without whom I would never have decided to pursue a degree in Computer Science. I took the course that is the topic of this dissertation with him and fell in love with programming. When I started teaching it, he provided me all of his material and supported my efforts every step of the way. He has spent countless hours talking with me about every issue having to do with the course and has been my rock of sanity when dealing with difficulties like students lying about plagiarism. He has also been as eager as I for answers to the questions that motivated this dissertation, and has enthusiastically supported this research, reminding me of its relevance and helping me to maintain my motivation.

I would like to thank the third member of my committee, Professor Minsuk Shim, for helping me to narrow the scope of my expectations and thus make this research possible. Before taking the quantitative research course with her, I could not see that any question I had was worthy of a dissertation. She showed me that this is the beginning, not the end, of research, which curbed my overwhelming anxiety and made this doable. She also is the primary advisor who helped me with experiment design

and reporting of results.

Finally, I would like to thank Professor Joan Peckham for allowing me to teach the object-oriented programming class for nine semesters, giving me plenty of opportunity to improve the course and see what changes seemed to help. I would also like to thank her for having me do advising for computer science majors early in the program. Time spent advising allowed me to better know my students and helped me to understand their vantage point on their performance in computer science courses. I would also like to thank her for providing me with the resources I needed to conduct this research: funding, hardware and software, acting as primary investigator for the IRB applications (including recruiting students), and last-minute signatures whenever I needed them.

Dedication

This is dedicated first to my husband Krzysztof, without whom I could never have pursued this degree and done this research. His unfailing emotional support has bolstered me during the long months of doubt and frustration. His logistical support has provided stability and has freed my time to do this work. His sense of humor has kept me sane.

This is dedicated second to my father-in-law, Dr. Frank Mathews. He has also provided emotional support from start to finish, allowing me to pursue this degree without constantly asking the question "why, what are you going to do with it?" He provided financial support that made it possible, not only because it helped pay the bills, but, more importantly, freed me of anxiety about financial decisions. As a hedonist following the original teachings of Epicurus, I believe the greatest good is pleasure, the greatest pleasure is freedom from pain, and the greatest pain is anxiety. Frank alone is responsible for giving me that freedom, and it is only because of that freedom from anxiety that I could stay in school and pursue this research.

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter 1

Introduction

Worldwide, and over time, the average successful completion rate in introductory computer programming courses has been 67.7% (Watson & Li, 2014). Furthermore, the percentage of Bachelor degrees in Computer Science granted to women has declined from 37% in 1986 to 18% between 2005/2006 and 2012/2013, the most recent year for which the DOE has data (National Center for Education Statistics, 2014). Either statistic alone would be concerning, but when you further consider that computer science has been in the top ten degrees for starting and midterm salaries (Payscale College Salary Report 2016-2017), the early bars to success and the low percentage of women graduates suggests an alarming systemic exclusion that needs to be addressed.

Several studies have been done that have concluded that the low number of women is a recruitment issue more than a problem with retention. Nonetheless, that high failure rate may contribute to the fact that few women consider majoring in computer science. If we can increase the rate of success, will that help in recruiting (and retaining) women?

There are many factors that contribute to success (or failure) in learning programming. Two of the studies that have been done looking at predictors of success have found that math is (unsurprisingly) a significant factor. No study has been done to look more closely at the relationship between math and learning programming.

What measure of math achievement can best predict success? What about math helps students to learn programming? Can math either help in identifying those likely to struggle or in fact help them to succeed? Do women with the same math achievement as men succeed in learning programming at the same rate?

Need for the study

In 2012 Andrew Hacker wrote an Op Ed article in the New York Times asking "Is Algebra Necessary?" In it he questioned the higher math prerequisites for majors that had no need for them. He did not question whether computer science should require math, but computer science students who struggle with math prerequisites have wondered whether those prerequisites are useful or merely gatekeepers. No study has been done to show what the relationship between math and learning programming is.

Two studies have shown that math is a predictor of success, but the question is what is it about math that helps in learning programming? How much math is useful? Does learning math in fact improve performance, or is it more a matter of being good at math?

This study seeks to address the needs of the student struggling to learn programming by looking specifically at conceptual predictors of success, focusing on math. If we can identify who is likely to struggle, we can offer remediation from the beginning of the semester in order to help them succeed. It also seeks to evaluate whether the math prerequisite for learning programming is, in fact, helpful.

Barriers to success

There are three potential barriers to success that this research is looking at. The first is the high failure rate in beginning programming. If we can identify who is vulnerable, we can provide remediations that will hopefully help them succeed. The second is asking whether math prerequisites are mere hurdles to overcome, or do they in fact help prepare a student to succeed in learning programming. The third is whether women are succeeding at the same rate as men. If not, identifying where they are not succeeding at the same rate may help detect barriers to their success at a specific institution.

Purpose of the study

The focus of this research is on identifying which conceptual factors lead to success in learning programming, and thus conversely, which students are more vulnerable to struggle in the course because they have difficulty grasping the concepts. By identifying the vulnerable population, we can help them early with remediations with the goal of helping improve performance in learning computer programming. In scrutinizing the relationship between math and programming we also can get a better idea of how much learning math helps in learning programming. Also, we wish to determine whether women with the same math achievement are succeeding in learning programming at the same rate as men. If not, then women may face other factors inhibiting their success.

Research questions and hypotheses

The first question is what measure of math achievement has the highest correlation to success in learning programming. The measures of math achievement to be tested are: grades in individual math courses taken during the year prior to the programming class; most recent math grade; average of the math grades of the year prior to the programming class; and the level of math achieved prior to the programming class. The hypothesis is that there will be some statistically significant correlation for each of these measures, but which has the highest correlation is to be determined.

The second question is whether learning math improves performance in learning programming. This question has two parts. The first part looks at struggling math students: for those who do poorly when taking a math course, does retaking it and learning the material the second time improve their performance in learning programming? The hypothesis is that, for struggling math students learning difficult math does improve performance in learning programming. The second part looks at students who are not struggling in math: for those students, does learning more advanced math improve their performance in learning programming? The hypothesis is that learning more math does improve performance in learning programming.

The third question is whether the measure of math achievement with the highest correlation to success in learning programming (question 1) can be used to detect discrepancies in performance in learning programming between men and women with the same math achievement. The hypothesis is that using this measure of math achievement, a difference in correlation to success in learning programming

between men and women would indicate an underlying discrepancy in performance of men and women with equal math achievement.

The fourth question is what are the predictors of success in learning programming. Which factors are most significant, and how much of the variance in grade do they account for? The factors to be examined are math achievement (to be determined in question 1), SAT math, SAT verbal, GPA, major and gender.

Chapter 2

Literature Review

Completion rate in introductory programming

While the problem of high failure rates in introductory programming (CS1) worldwide has been acknowledged for some time, few studies have been done to measure failure rate. Bennedsen and Caspersen (2007) surveyed the authors and panel participants of five CS educational conferences. They had a 12.3% response rate (N = 63) from 15 different countries, and found that the worldwide mean of students passing was 67%, with a wide variance. In general they found that the pass rate was higher for smaller classes and for colleges rather than universities, but that the programming language used did not matter.

Watson and Li (2014) did a longitudinal study by searching articles published between 1960 and June, 2013 to find those that reported data on failure rates. In all they found 54 articles that described failure rates in 161 CS1 courses at 51 institutions across 15 countries that spanned from 1979-2013. The worldwide mean for passing was 67.7%. In neither study was there a common definition of "passing," whether anything above an F, or only those grades that allowed a student to continue to the next course (often a C or better), and whether the passing rates counted course attrition as well as failure. However, the means in the two studies being as close as they were presents a good argument that the population mean is around 67%.

Predictors of success in learning programming

In order to improve the passing rate for CS1, we need to have a better idea of what factors contribute to the high percentage of failure. One possibility is that the conceptual content underlying programming is too abstract for some students to fully grasp. Barker and Unger (1983) developed a tool to predict success in a CS1 course based on Jean Piaget's intellectual development (ID) levels. The concrete level is characterized by the use of logic applied to concrete problems. It involves inductive reasoning, but not deductive reasoning. The formal level is characterized by hypothetical and deductive reasoning and the ability to use symbols related to abstract concepts in a logical way. Barker and Unger's instrument had 11 questions that were categorized as concrete, early formal, formal or late formal. Answering both early formal questions (direct proportion and probabilistic reasoning) incorrectly placed students in the late concrete category. If either was answered correctly the student was placed in early formal, and if, in addition, the student answered three out of four of the late formal questions (propositional and correlational reasoning, deductive logic or permutations) they were categorized as late formal. Using an Anova test they found statistically significant differences at the .05 level of significance between the grades on the exams by ID levels assessed using their tool. This suggests that those who struggle with abstract thinking may need more time to internalize the concepts upon which programming depends.

Other research has been done to find predictors of success in programming. In response to a high demand for the course that the faculty could not meet, Leeper and Silver (1982) sought a means of filtering out students less likely to succeed in their

program. They found some correlation between math and verbal SAT scores and success, but these along with the other factors they considered (rank in high school and grades in math, English and language) accounted for at most 25% of the variation of grades.

Wilson and Shrock (2001) tested a model with 12 predictive factors, including math background, attribution for success / failure (explanations students give for their success or failure on the midterm exam), domain-specific self-efficacy, encouragement, comfort level in the course, work style preference (competitive or cooperative), previous programming experience, previous non-programming computer experience and gender. Their sample was 105 students who volunteered for the study out of 130 taking a CS1 course during Spring 2000 at a midwestern university. They used two validated surveys to gather the data for the independent variables and midterm grades rather than final grades so they could include students who drop the class. [Midterm grades had a very high correlation (Pearson $r = .97$, $p < .001$) with final grades.]

In their general linear model they found that the 12 factors contributed to 44% of the variance among the grades ($F(12, 92) = 6.13$, $p < .001$), but using a stepwise linear regression they found that five of the factors accounted for 40% of the variance. Comfort level, math and a competitive work style preference positively correlated with performance on the midterm, while attribution of performance on the exam to luck or the difficulty of the task negatively correlated with performance on the midterm. Furthermore, while prior programming experience in general did not show any effects, they found a prior formal class in programming to be predictive of

success. And while other computer experience (internet, games, office applications) in general did not have any effects, hours playing computer games did have a negative influence.

Simon, Fincher, Robins et. al. (2006) tested a different set of predictors: spatial visualization and reasoning; designing and sketching a map; articulating a search strategy for finding a name in a phone book; and attitudinal factors, to see which correlated with success. Their research is noteworthy both because their sample (177 students) came from 11 institutions and because of the creative design of the instruments. However, some of their correlations are statistically significant, though not strong. Others are not statistically significant unless they also include the students who did not complete the course. They found a trend toward students who created survey maps that modeled both the routes and the landmarks to be stronger at programming than those who sketched out routes or landmarks alone. They also found students who could better articulate their search strategy to do better than those who were less articulate.

In general their predictors are analogs to programming, so it is unsurprising that someone who can create a more complete abstract model or who can articulate their search methodology in more detail shows better performance in problem solving in a domain involving abstraction and algorithmic thinking.

Motivation and habits of students who failed

Sheard and Hagan (1998) looked at differences between students who were taking the course again and new students. Their introductory programming course is

generally taken first semester freshman year, and the first semester has an enrollment of 400 students. The second semester has an enrollment of 80 students. The study was done second semester 1997, where of the 84 students enrolled, 58% were taking it for at least the second time. They did a survey halfway through the semester and another at the end of the semester. They found that most of the repeat students had little interest in programming, but had wanted to get into business school and information technology was as close as they could come. The repeat students also worked significantly more hours at jobs outside school than the students new to the course. Many of the repeat students had poor attendance at lectures. The authors described the repeat students as having a shallow learning approach, being reluctant to seek out and explore extra resources at their own initiative. Many of the repeat students did not use or own the textbook, though it was strongly recommended. While about a quarter of the new students failed the course, over a third of the repeat students failed the class again. This study suggests lack of motivation to learn programming is the primary issue, but since the study was done after the students had failed the course for the first time, their lack of motivation and interest in other majors may in some cases be a result rather than a cause of their failing the first time.

Gender gap

Previous research on the gender gap in Computer Science has concluded that it is primarily due to problems with recruitment caused by negative stereotypes girls have of the field. In 2009 Cohen and Deterding published a study based on national data from 1999-2003 of students entering or declaring engineering fields (freshman

and sophomore years) compared to the number of degrees granted in each field. The study looked at retention of women versus men in two ways and determined that nationally the retention of women was the same as men.

In 2015 Cheryan et al. published an article detailing how stereotypes in computer science steer girls away from the field from a young age. As one example of her research on recruitment, Cheryan exposed Stanford students who were not majoring in computer science to one of two rooms: one decorated in a stereotypical way (Star Trek posters, science fiction books and soda cans) and the other decorated in a non-stereotypical way (nature posters, neutral books, water bottles). The women exposed to the non-stereotypical room expressed significantly more interest in the major than those exposed to the stereotypical room.

Much work has also been done on retention. The work on retention focuses on increasing comfort and a sense of belonging. As one example, Werner et al. (2004) found that pair programming improved retention for women by changing programming from a solitary activity to a collaborative one.

Conceptual factors

This research is focused primarily on the more conceptual factors that contribute to success in programming. The goal is twofold: to identify conceptual weaknesses that may be inhibiting students from understanding programming, and to determine whether for students who do not have those weaknesses, women are succeeding at the same rate as men. If women are succeeding at a lower rate than men, that would suggest that comfort level, one of the factors found in previous research to

contribute to success, may very well be an issue.

Another of the factors previous research identified as contributing to success is a preference for a competitive work style. This was found in a study of one class with an enrollment of 130 students. Given that the teacher, grading scale and/or course expectations may have favored those who prefer to compete, it is not clear whether that result can be generalized to anyone desiring to learn computer programming. This may be a reflection of the culture of computer science more than a trait that helps students to learn how to program.

Some of the research has been to identify who is likely to be good at programming. The study that used drawing a map and articulating a search strategy as predictors of success identified students who would succeed in solving problems by their showing that they already knew how to solve very similar problems. The current research is to determine what the underlying conceptual traits are that help students succeed. As a step after identifying these factors, it would be interesting to see if *teaching* students to draw better maps and better articulate a problem-solving strategy is an effective remediation strategy for students who are vulnerable.

Several studies found non-conceptual factors that negatively contributed to success. Those include attributing performance on an exam to luck or the problems being too hard, spending a lot of time playing video games, not being interested in learning programming, spending a lot of time out of class working at a job, not reading (or owning) the text book and poor class attendance. Lack of interest and not spending the necessary time on the material are typical reasons to do poorly in any course. We can assume that a portion of the variance of the grade has to do with motivation, but

the goal of this research is to help those who are motivated, but having a difficult time grasping the material.

This research is more closely related to the study that found a correlation between abstract thinking and success in learning programming, and the studies that found that math was a positive factor. In this research I am looking more closely at the correlation between math and programming on the belief that the abstract thinking that underlies success in learning math is similar to that of learning programming.

Chapter 3

Methods

The questions this research is working to answer are does learning higher math improve the ability to program? Is there a particular math course that best prepares students for programming? Can the correlation between math achievement and programming be attributed primarily to math aptitude? How much does the ability to do math contribute to the ability to learn programming? How much of a role does verbal aptitude play? How much of success can be attributed to academic achievement? And, finally, do women with the same math ability as men succeed in learning programming at the same rate?

Definitions

In 1978 the ACM's Computing Curricula developed the terminology "CS1" and "CS2" to refer to the introductory programming course (CS1) and the course on data structures and abstraction (CS2). While there is no consensus any longer on which topics should be covered, the goals of CS1 are that students should come away with a working knowledge of software design issues and software design methodology, algorithm design and analysis (at an introductory level), problem solving strategies, abstraction and be familiar enough with the syntax of some language that they can implement solutions to problems following design principles (Marion, 1999). While the object-oriented programming class is not the first programming course for Computer Science majors at the University of Rhode Island,

it is the first one whose emphasis is on software design, and is the first one that introduces abstraction. For Computer Engineering students this is the introductory course in structured programming. (Their prior experience is a brief introduction to assembly code.) Therefore object-oriented programming is the closest to a traditional CS1 course like those in the studies cited above. In the rest of this paper "CS1" will be used to refer to the object-oriented programming class.

This study is to determine predictors of success in learning programming. For this we need to define conditions of success. There are two populations who take CS1: Computer Science and Computer Engineering majors, who must also succeed in CS2, and other majors who may or may not take CS2. Since the majority of the students taking CS1 are Computer Science and Computer Engineering majors, for this study success in CS1 means likely success in CS2 as well. The minimum condition of success, therefore is passing both CS1 and CS2 the first time with a grade of C- or better in both. The reason for this is that passing with below a C- does not show that the material was truly learned, and needing to retake a course also indicates that the material was not adequately learned the first time.

Part 1, what is the relationship of math to success in learning programming?

This research consists of two parts, both ex post facto. The first part is looking at the relationship between math and programming, and has three research questions. The first question is what is the measure of math achievement with the highest correlation to final grade in CS1 (the first time taken). The second question is, assuming success in math does correlate to success in learning programming, does

learning math improve performance, or is being good at math regardless of learning higher math sufficient. The third question is do women with the same math achievement as men have the same rate of success in learning programming.

Research design and rationale

For all of the questions in part 1, the data are the grades in CS1 and math. The first question, determining the measure of math achievement with the highest correlation to success in learning programming, is a comparative correlational study looking at math achievement in a number of ways to find the measure that best predicts success in CS1.

The second question, does learning math improve performance in CS1, will be answered in two ways, both comparing means. The first determines whether there is a difference in grades in CS1 between those who take a math course but don't do well, those who do well the first time, and those who don't do well the first time and then retake the math course and improve their grade. The second determines whether there is a difference in grade in CS1 between those who take only precalculus and those who take precalculus and also succeed in calculus. It also determines whether there is a difference in grade in CS1 between those who take only calculus 1 and those who take calculus 1 and also succeed in calculus 2 or linear algebra.

For the third question, is there any difference in performance between women and men with the same level of math achievement, the methodology will be a comparative correlational study determining if, using the measure determined by the first question, there is any difference in the correlation between math achievement and

success in CS1 for men and women.

The reason for identifying the measure of math achievement with the highest correlation to success in learning CS1 is primarily for prediction. This measure could be used to identify both those most likely to struggle and those most likely to do very well. It would be useful to identify those most likely to struggle so we can provide remediation to help them to better grasp the material. It may be useful to identify those most likely to do very well either to fast-track students into the course or to offer an advanced version of the course that may cover material of interest to the best students that is not covered in the course currently.

The reason for determining whether learning math improves performance in programming is to evaluate math prerequisites or to determine if taking any particular math course may be helpful. The rationale for a comparative means test is that it allows us to compare two groups of students who both struggle with a math course, one of which goes directly into CS1, the other of which improves their math grade prior to taking CS1. If there is a statistically significant difference in the mean grade in CS1 for those two groups, this suggests that where math ability is the same, learning the material makes a difference. The rationale for comparing performance between those who have only had precalculus (or calculus) and those who have successfully completed calculus 1 (or beyond calculus 1), is that if there is little to no difference, that may suggest that being good at math is what helps people to learn programming, not learning math. Conversely, if there is significant improvement, that may suggest that learning programming improves with learning higher math.

The reason for learning whether there is a difference between the performance

in CS1 of men and women with the same math achievement is to identify any issues in our program with retention of women. The reason for comparing the measure of math achievement with the highest correlation to success to determine if it's the same for men and women is that this measure should be most sensitive to different success for the same math achievement. Identifying any difference between women and men would point to where further research should be done to find why the percentage of women graduating with CS degrees is so low compared to men.

Sample

The population are college students taking CS1 at a medium sized state school in northeastern U.S.A. These students are Computer Science, Computer Engineering, Math or Physics majors and others interested in learning how to program, or taking it as a prerequisite for the graduate program in Computer Science. For the first part, the sample is all students who took CS1 between spring 2005 and fall 2015 (approximately 1100 participants). During this time there were primarily three instructors and the course was taught using Java.

Data Source

All data for the first part comes from the office of enrollment services. The data includes grades and terms for math and computer science courses, major at the time CS1 was taken and gender. The dataset was classified as exempt by the head of the IRB.

Research question 1: measure of math achievement

Variables of interest

The dependent variable for all correlation tests is the grade for CS1 the first time it is taken, assuming that first time is not a transfer, withdrawal or no work. In the case that a student transfers CS1 and then retakes it, the grade received when taken at the institution will be used as the first grade. Likewise, if a student withdraws or does no work the first time, the grade the second time the course is taken will be used as the first grade. For all tests it is assumed that the grade in CS1 is normally distributed, and this assumption will be checked.

Precalculus is a prerequisite of CS1. Most students taking CS1 have thus taken precalculus or calculus 1, and some have already completed calculus 2 and/or linear algebra. The raw data in calculating each of the variables of interest are the course grades in these four courses and when the course was taken in relationship to taking CS1. The variables of interest are:

- Level of math achieved prior to taking CS1
- Grades in individual math courses taken the year prior to CS1
- Most recent math grade prior to CS1
- Average of all math grades taken in the year prior to CS1

For level of math achieved, the levels are:

  - -1 – less than precalculus
  - 0 – no information on math level prior to CS1
  - 1 – precalculus

- 2 – calculus 1 (applied calculus or intro. calculus and analytic geometry)
- 3 – beyond calculus 1 (calculus 2 and/or linear algebra)

The grades considered in determining the levels are either in semesters prior to CS1 or, in the case of transferred math grades, the same semester as CS1. For the difference between less than precalculus and no information, precalculus grades received the same semester or after CS1 are used. In order to be classified as level 1 or above, the grade in the course must be a C- or higher. Someone who has passed calculus 1 prior to CS1, but has not earned a grade of C- or higher, are classified as a 1. Someone who started with calculus and fails is classified as a -1 along with those who take precalculus but get lower than 1.7 and those who take precalculus at the same time or after CS1.

For grades in individual math courses, the courses to be used are precalculus, calculus 1, calculus 2 and linear algebra. For those who have taken the course multiple times prior to CS1, the last grade will be used. For this and any other measure using grade information, grades that reflect dropping, transferring or doing no work for the course will not be used. In the case of transferred grades, we have no information about how well the student in fact did, only that they achieved a C or better. In the case of dropped or no work grades, the student did not take the course.

For most recent grade, only courses taken within the year prior to CS1 are used. Since precalculus and calculus 1 are prerequisites for calculus 2 and linear algebra, the only two courses that could be taken simultaneously are calculus 2 and linear algebra. If a student does take those two courses at the same time within the year prior to taking CS1, the average of the two grades is used. Otherwise, the grade in

the course taken closest to CS1 is used. If that course was taken multiple times prior to CS1, the last grade is used unless it is a withdrawal or no work, in which case the first grade is used. If the first grade is a withdrawal or no work, the previous class is considered the most recent. If the most recent grade is a transfer, including if math is transferred the same semester as CS1, it is not included.

For the average of all math grades taken prior, all grades in any of those four math courses taken in the previous year (not including transfers, withdrawals or no work) will be used to calculate the average. This average will be used for all students who have taken a math course at the institution in the year prior to CS1, including those who may have transferred courses after those on which the average is based.

In order to use whatever measure has the highest correlation, we need to determine the threshold of success. To do so, we define categories of success. These categories are:

0. not passing CS1
1. not taking CS2
2. not passing CS2
3. passing both, but either with at least one D or retaking at least one
4. passing both first time with C- or better (includes those transferring CS2)
5. passing both first time with A's or B's

The independent variables used to calculate the category of success are grades in CS1 and CS2.

Data analysis

To determine the measure of math achievement with the highest correlation to success in CS1, the correlations between each variable of interest and the first grade for CS1 will be compared. After determining the measure with the highest correlation, a linear regression will be done using this measure and other factors like gender and major to determine how much of the variance in the grade it accounts for.

In addition to determining how significant a factor math achievement is in success in CS1, in order to use a measure of math achievement to predict success, we need a threshold below which students are more likely to struggle and above which more likely to succeed. The first step is determining the grade threshold in CS1 above which students are more likely to succeed in both CS1 and CS2. In order to do this, the level of success is determined for all Computer Science and Computer Engineering majors, based on performance in CS1 and CS2. Then for each grade in CS1 the percent of students that fall into each category of success is calculated. The threshold is based on the grade in CS1 below which the percent of students successful in both CS1 and CS2 (categories 4 and 5) drop significantly.

Once the CS1 threshold is determined, it is used to determine what value of the measure of math achievement with the highest correlation to CS1 grades has a mean closest to that threshold. That value is the threshold of math achievement below which students are more likely to struggle in CS1.

Research question 2: does learning math improve performance in learning programming?

Variables of interest

- precalculus/calculus competence
- precalculus / calculus pair
- anticipated CS1 grade
- category of actual vs anticipated CS1 grade

For competence, only those who have taken precalculus (or calculus) in the year prior to CS1 and have not had more math beyond precalculus (or calculus) prior to CS1 will be included. Grades reflecting no work, withdrawal or transfer are not included. The first and last grades are used to determine whether improving one's grade makes a difference. The categories are:

1. taking precalculus/calculus once and getting less than a C- or taking it two (or more) times prior to taking CS1 and getting less than a C- the first time and not improving to at least a C by the last time,
2. taking it more than once prior to CS1 and getting lower than a C- the first time and at least a C the last time,
3. getting a C- or better the first time

For precalculus / calculus pair, the goal is to see if we compare two students, one of whom has taken only precalculus and the other of whom has taken precalculus and calculus, if there is a difference in performance in CS1. In order for this to be

meaningful, the students who are paired must be similar enough that any difference in grade is likely due to having more math. In order to be as similar as possible, subjects are paired on the basis of having the same grade in precalculus, the same major and the same gender. The subject who has both precalculus and calculus must also have achieved at least a 1.7 in calculus. For both subjects, if precalculus was taken more than once prior to CS1, the last grade before CS1 will be used in assigning the pairs. In order to limit variance due to instructor, all subjects will have taken CS1 with the same instructor. The pair variable is the CS1 grade of the subject to whom they are paired.

The same pairing will be done with those who have had calculus and those who have successfully completed either calculus 2 or linear algebra.

The anticipated CS1 grade will be determined using grades of students who took precalculus (or calculus) in the year prior to CS1 and did not take another course beyond precalculus (or calculus) prior to taking CS1. The anticipated grades are means of CS1 grade of the precalculus (or calculus) grades of that group. Grades of withdrawal, no work or transfer will not be used in the generation of the anticipated grades. For each precalculus grade an anticipated CS1 grade will be generated (11 values from 0.0 to 4.0).

For the category of actual versus anticipated CS1 grade, these will be calculated both for those who only had precalculus (or calculus) prior to CS1 and for those who had precalculus (or calculus) and also passed calculus (or a course beyond calculus) with at least a C- prior to CS1. These will be based on the anticipated grades in CS1 and the actual grades in CS1. There will be three categories:

- -1: doing over a grade worse than anticipated CS1 grade

- 0: doing within a grade of anticipated CS1 grade (in either direction),
- 1: doing over a grade better than anticipated CS1 grade.

Data analysis

The first test is a comparison of means to see if learning precalculus or calculus causes a difference in grades in CS1. This is an Anova test using precalculus (or calculus) competence as the independent variable with the grade in CS1 as the dependent variable. This compares the performance of three groups of students who took precalculus (or calculus) most recently: those who did poorly, those who did well the first time, and those who did poorly and retook the course and improved their grade prior to CS1. If there is a statistically significant difference in the means of the grades for the students who did poorly and retook the course, doing better the second time, than those who did poorly and didn't retake the course (and those who did poorly more than once), that indicates that, at least for students who struggle in math, learning math does improve the ability to learn programming.

The second test is a paired sample t-test comparing the CS1 grades of those who have gone beyond precalculus (or calculus) with their precalculus (or calculus) pair. If there is a significant difference in means between the two groups, that indicates that learning more math has an effect on learning programming.

The third test, used to assess the effect of succeeding in the next math class for everyone who has gone beyond, has two parts. The first part determines the anticipated CS1 grade and the second part uses that value to determine what percentage of those who go beyond show significant improvement. The first part uses

different groupings of the precalculus (or calculus) grades to find means of CS1 grades so that each mean is not the mean of one math grade alone, but is the mean of that grade, the grade above and the grade below. The goal is not to find an exact mean, but to generate a set of means that increase with the increasing math grades. Only the grades of the students who have taken precalculus (or calculus) in the year prior to CS1 and did not take more math beyond precalculus (or calculus) prior to CS1 will be used in determining these means. Then the precalculus (or calculus) grades are used to assign the associated means to the anticipated CS1 grade for that student. The anticipated CS1 grades will be assigned for both the students who did not go beyond precalculus (or calculus) and those who did.

In the second part, the difference between the actual grade and the anticipated grade will be used to assign the category of actual versus anticipated CS1 grades. The reason for using over a full grade below and over a full grade above is to insure that the mean is within the largest 95% confidence interval of the means generated. The comparison in this test is not to see how many students have grades below, above or the same as the anticipated grade. The comparison is between the distribution of the categories for the control group (those whose grades are used to generate the means) and the experimental group (those who have successfully completed a course beyond precalculus or calculus). A chi square test to see if the proportions in each category are different for the control and experimental groups will be used to determine whether there is an effect of learning more math.

Research Question 3: Do women with the same math achievement as men have the same success in learning CS1?

Variables of interest

The dependent variable is grade in CS1.

The independent variables are:

- the measure of math achievement from research question 1
- gender

Data analysis

The measure of math achievement with the highest correlation to success in CS1 will be used to separately calculate the correlation for men and for women. If there is a difference, we'll look more closely at the grade distributions.

Part 2, factors of success in learning programming

Research design and rationale

Part 2 of the study works with a smaller dataset with information not available in the larger dataset used in part 1. This is a causal comparative analysis to find predictors for success in learning programming. The reason for looking for predictors of success is to improve our ability to predict who is more likely to struggle. Specifically this study is looking at factors associated with conceptual understanding and academic achievement. These are measures that may be available to an advisor

and could be used to insure that someone likely to struggle could benefit from remediation starting at the beginning of the semester.

The design is causal comparative in order to assess how much each factor contributes to success in learning programming. This will identify both major and minor factors, and which measures make no meaningful contribution to the variance in grade.

Variables of interest

The factors to be tested are:

- SAT math scores
- SAT verbal scores
- GPA before taking CS1
- math achievement (the measure of which is determined by the first part)
- major
- gender

and the dependent variable is:

- grade for CS1

The College Board research studies validated the SAT scores as a very strong predictor of first year college grade point average (Validity Studies, 2017). Writing is the most predictive section of the SAT, slightly more predictive than either math or critical reading. SAT math scores are a widely used measure of math achievement from high school.

One of the previous studies (Simon, 2006) found that those who were better able to verbally articulate a problem-solving strategy were more successful in learning how to program. This research is including SAT verbal scores as one of the factors, since they are a widely used measure of critical reading, which involves reading comprehension and recognizing nuance.

GPA is a measure of overall course performance, and as such is important to consider in the performance in CS1. The GPA is from the semesters up to, but not including, the semester CS1 was taken.

The measure of math achievement is from part 1 of this study, assuming that a measure with predictive power is identified.

Major is either computer science, computer engineering or other.

The grade for CS1 is the grade received the first time CS1 was taken. If the first time is a withdrawal, no work or transfer, the second time the course is taken is used as the first.

Sample

The sampling is of volunteers among students taking CS1 during the Fall 2015, Spring 2016 and Fall 2016 semesters (approximately 100 participants). The first two semesters had one instructor while the third semester had a different instructor who, while experienced, was teaching CS1 for the first time.

The recruitment of the volunteers was done by someone other than the instructor, with the instructor not present. The recruiter explained that the role of participants would be to consent to allow their data to be used. For Fall 2015 and

Spring 2016, the sampling is 40% of those enrolled in the class. For Fall 2016 the sampling is 48% of those enrolled in the class.

Data source

All data for the second part comes from the office of enrollment services at the institution.

Data analysis

The analysis uses a linear regression with all of the factors above and CS1 grade as the dependent variable. The regression will be done with blocks to see the change in R^2 with the inclusion of each factor.

In addition, a correlation test will be done using the measure of math achievement and the CS1 grade to see if the correlation is consistent with the findings in part 1. Also, a correlation test will be done between SAT math scores and the measure of math achievement. If the correlation is high, SAT math scores may be used instead of the measure of math achievement when that measure is not available.

Chapter 4

Results

Part 1 data

There were 1122 students who took CS1 between 2005 and 2015, inclusive. Of those, 148 students (13.2%) transferred the class and did not retake it at the institution. Of the 974 students who did not transfer the course, 42 (4.3%) dropped the class or did no work one or more times, never truly taking the class. This number is bolstered by the fact that it includes students who dropped or did no work during 2015, when they may have retaken the class later. Considering only those who took CS1 at the institution prior to 2015, of the 832 students, 31 (3.7%) dropped the class or did no work one or more times. (The percentage who dropped is likely higher, since the university did not record withdrawals prior to the 2013 academic year.)

Of the 932 students who took the course, the distribution of the grades (Figure 1) is:

- 30.5% A-range,
- 26.4% B-range,
- 19.5% C-range,
- 7.9% D-range and
- 15.7% failing.

The first thing to note is that the grades are not normally distributed, but are heavily weighted towards A's and B's. Although the grade distributions of the instructors are not the same, none are normal and all are heavily weighted towards A's

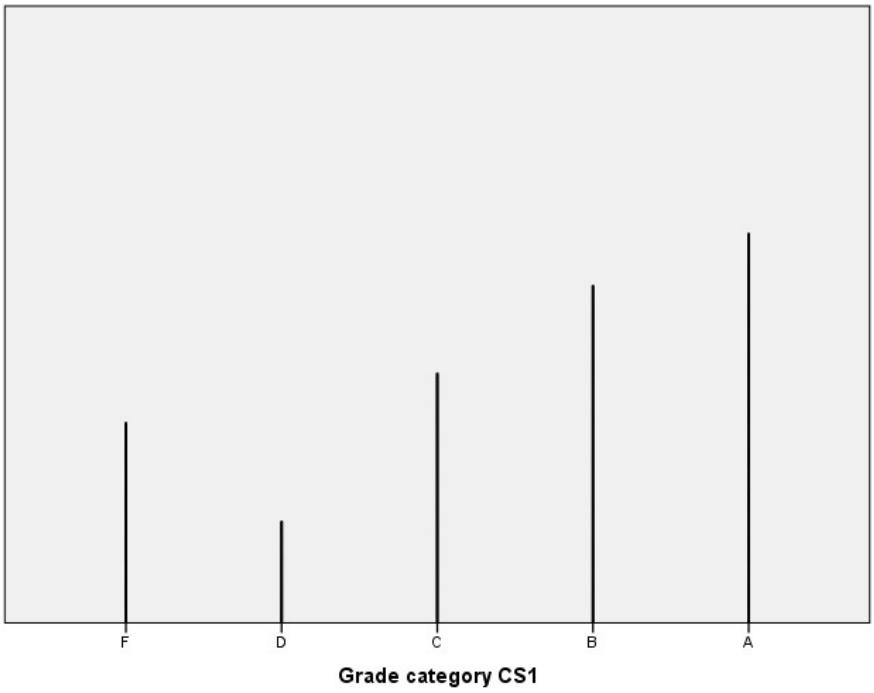


Figure 1: Grade distribution for CS1

and B's. It is worth asking whether these high grades are a result of grade inflation. To determine this, we look at how well students do in CS2. Not all students who do well in CS1 need to take the following course, but it is required for all Computer Science and Computer Engineering majors (the majority of the students). In Figure 2 the population are all CS and CE majors who took CS1 for the first time at the institution prior to 2015.

The great majority of those who received A's in CS1 received an A or B in CS2, and almost all took CS2. Approximately 2/3 of those who received a B in CS1 received a C or better in CS2. The majority of students who did well in CS1 received grades in CS2 that are within one grade of what they received in CS1. Approximately 2/3 of those who receive a C in CS1 either do not continue or receive a D or F the first

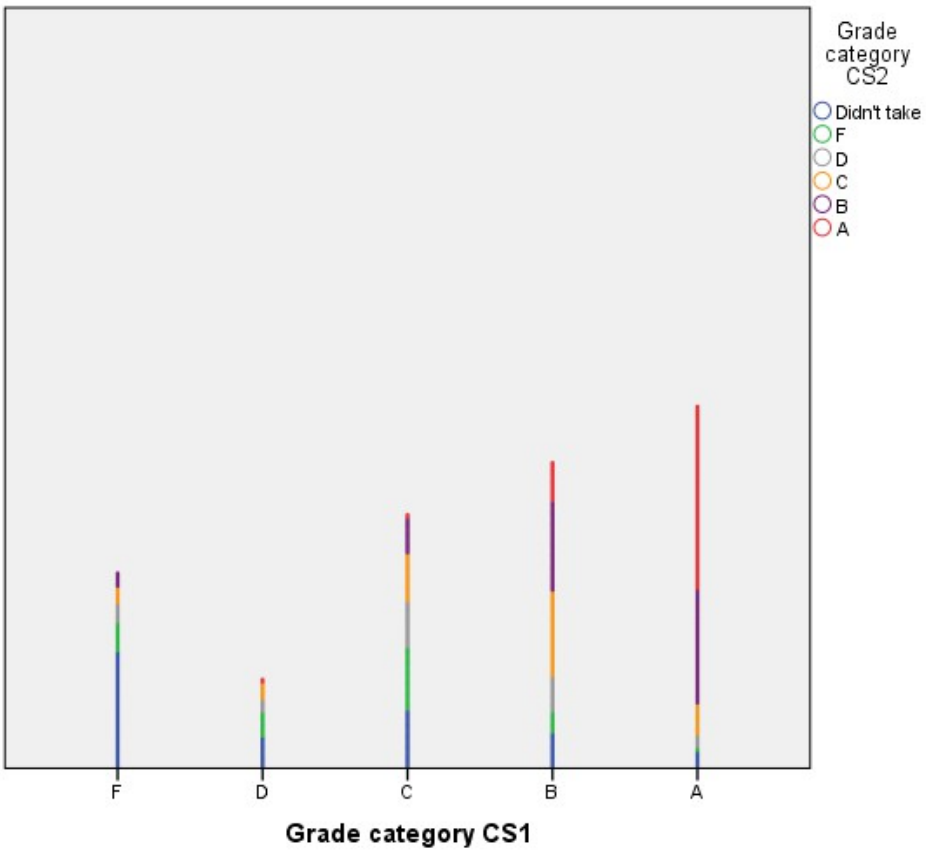


Figure 2: Grades for CS & CE majors

time they take CS2. Most of the remaining 1/3 receive a C. The great majority of those who receive below a C in CS1 either do not take CS2 or receive a D or below the first time they take CS2. This suggests both that the grades are reasonably consistent (not a result of grade inflation) and that doing well in CS1 predicts success in CS2.

The grade distribution, corroborated by the grades in the following course, has implications for the research questions. All tests assume a normal distribution of the dependent variable. The further away from a normal distribution, the less valid the results of any of the tests will be. Tests for normality, both Shapiro-Wilks and Kolmogorov-Smirnov, pass (N=932, p<.001), so we reject the null hypothesis that there is no difference between this distribution and normal. The Q-Q plot (Figure 3)

indicates a heavier concentration in the left tail (grades of F).

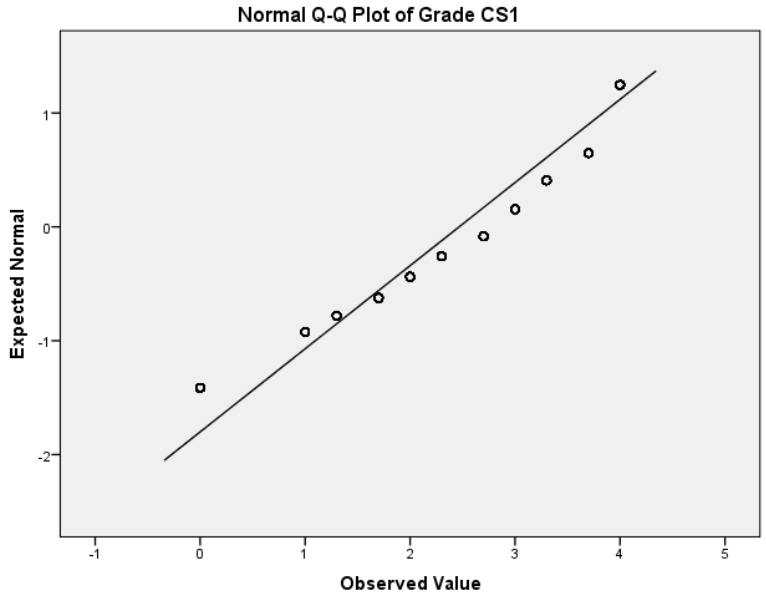


Figure 3: Normality plot of CS1 grades

Transformations to normalize the CS1 grades (logarithmic, square root and exponential) did not improve the normality of the distribution. Tests for normality, both Shapiro-Wilks and Kolmogorov-Smirnov again pass (N=932, p<.001), so we must reject the null hypothesis that there is no difference between these distributions and normal.

The not normal grade distribution may reduce correlations between the math measures and CS1 grades. For this reason, in addition to determining correlations, we also want to consider the threshold for success to see whether there is a threshold below which students are far less likely to succeed, above which more likely to succeed.

There were 580 students who were CS or CE majors who took CS1 at the institution before 2015 (so the dataset includes their CS2 grades). Of those, 50.7%

passed both courses with a C- or better the first time (Table 1 & Figure 4). Those who receive a B or better tend to do well in the succeeding course, those who receive a D or below tend not to (or pass with D's) and those receiving a C are split, with 2/3 not taking or doing below a C and 1/3 getting a C in CS2. When we look more closely at those receiving a C, the threshold appears to be between C and C+. Half of those receiving a C+ in CS1 receive a grade in the C-range or above the first time in CS2, while less than a quarter of those receiving a C or C- receive a grade in the C-range or above the first time in CS2. There were 369 students who got a C+ or better in CS1. Of those, 75% passed CS2 the first time with at least a C-. Of the 211 students who got a C or below in CS1, only 8% passed CS1 & CS2 the first time with at least a C-. We are thus using C+ as the threshold for success in CS1.

That threshold may be higher than necessary for non-majors, since success does not depend on also succeeding in CS2. There were 249 students who were not CS or CE majors who took CS1 at URI. Of those, 194 (77.6%) received a grade of C- or

	Frequency	Percent
not passing CS1	54	9.3
not taking CS2	77	13.3
not passing CS2	31	5.3
passing both, either with D in at least one or retaking at least one	124	21.4
passing both first time with C- or better	93	16.0
passing both first time with A's or B's	201	34.7
Total	580	100.0

Table 1: Success distribution for majors

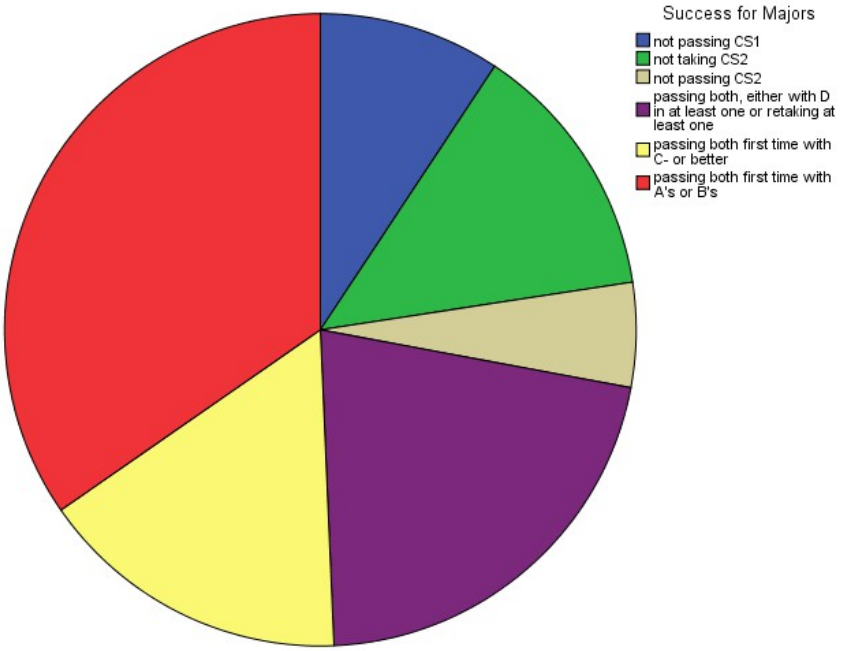


Figure 4: Pie chart of success distribution for majors

	Frequency	Percent
not passing	36	14.5
D	20	8.0
C	34	13.7
A or B	159	63.9
Total	249	100

Table 2: Success distribution in CS1 for non-majors

better (Table 2). While non-majors had higher performance overall than majors, using the higher threshold may help identify the non-majors more likely to find learning programming challenging.

To identify those likely to get an A or B in both courses, of those who earn an A in CS1, 89.2% get an A or B in CS2. Of those who earn an A-, 66.7% get an A or B

in CS2. The percentage drops below 50% after that. When identifying those likely to do very well, a threshold that would predict strong success for most would be 3.7.

Research question 1

Performance in specific math courses

The correlations between recent math grades and CS1 (Table 3) is generally moderate. While applied calculus 1 and linear algebra have the highest correlations with grades in CS1, since the number taking these courses is low, the course that a more substantial proportion of students are likely to take with the highest correlation is introductory calculus and analytic geometry (calc 1).

	Pearson's *R*	Significance	*N*
precalculus	.47	<.001	218
applied calculus 1	.74	.014	10
introductory calculus and analytic geometry	.52	<.001	343
calculus 2	.45	<.001	223
linear algebra	.57	<.001	47

Table 3: Correlations between course grades and CS1

When we look at the correlations for precalculus, calc 1, calc 2 and linear algebra taken in the previous year by major (Table 4), we see that for all but calculus 2 the correlation is higher for Computer Science majors than Computer Engineering or other majors. For Computer Engineering majors the correlation between calculus 2 and CS1 is highest, and for other majors any course but precalculus has a similar correlation. All of the statistically significant correlations are moderate.

	Comp. Science			Comp. Engineering			Other		
	P's *R*	Sig	*N*	P's *R*	Sig	*N*	P's *R*	Sig	*N*
precalculus	.49	<.001	169	.36	.038	34	.23	.406	15
intro. calculus & analytic geometry	.53	<.001	175	.49	<.001	119	.48	.001	49
calculus 2	.43	.002	48	.49	<.001	128	.45	.001	47
linear algebra	.65	.001	22	-.35	.772	3	.53	.012	22

Table 4: Correlations between course grades and CS1 by major

Pearson's correlation is linear, so it is important to confirm that some other test of correlation is not more appropriate. In looking at the scatter plots for precalculus, calculus 1 and calculus 2 (Figures 5-7), it does not appear that there is a non-linear pattern to any of them.

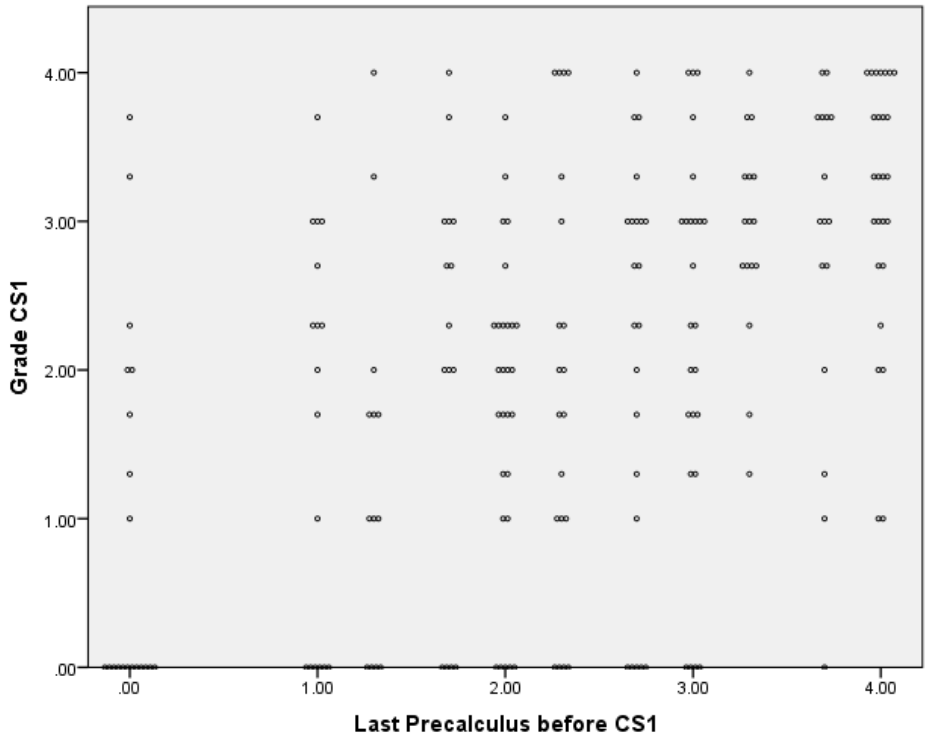


Figure 5: Scatter plot of correlation of precalculus and CS1 grades

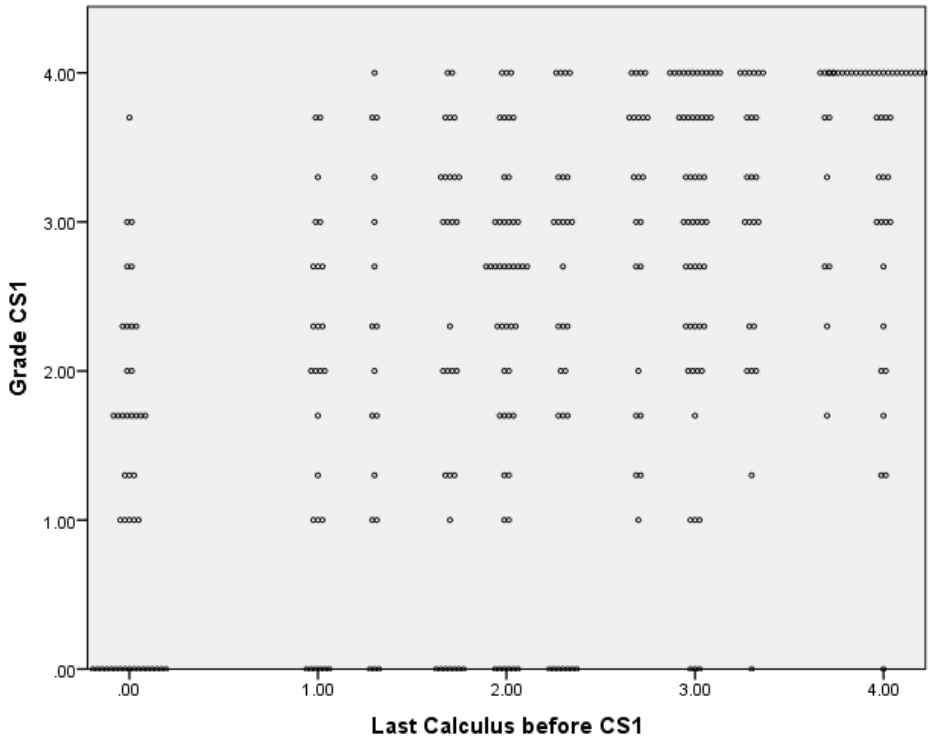


Figure 6: Scatter plot of correlation of Calc 1 and CS1 grades

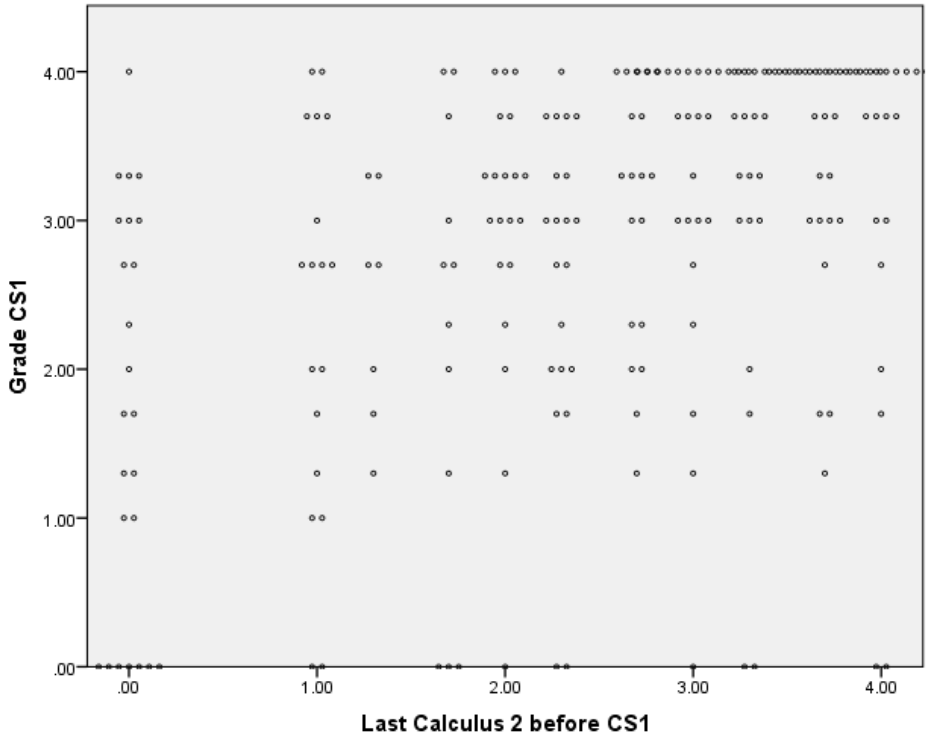


Figure 7: Scatter plot of correlation of Calc 2 and CS1 grades

Level of math

Since there are only four levels, it is not surprising that the correlation between math level and CS1 grades is moderately low, $r(700) = .41$, $p < .001$. The scatter plot (Figure 8) shows that there does not appear to be any non-linear correlation.

Using an Anova test to compare the means of the grades in CS1 (Table 5), the main effect of math level is significant at the $p < .05$ level: $F(3,698) = 48.37$, $p < .001$. Post hoc comparisons using the Tukey HSD tests show there is a statistically significant difference between all four levels[1]. For the difference between precalculus and calculus 1, $p = .019$, and for all other differences between means $p < .001$.

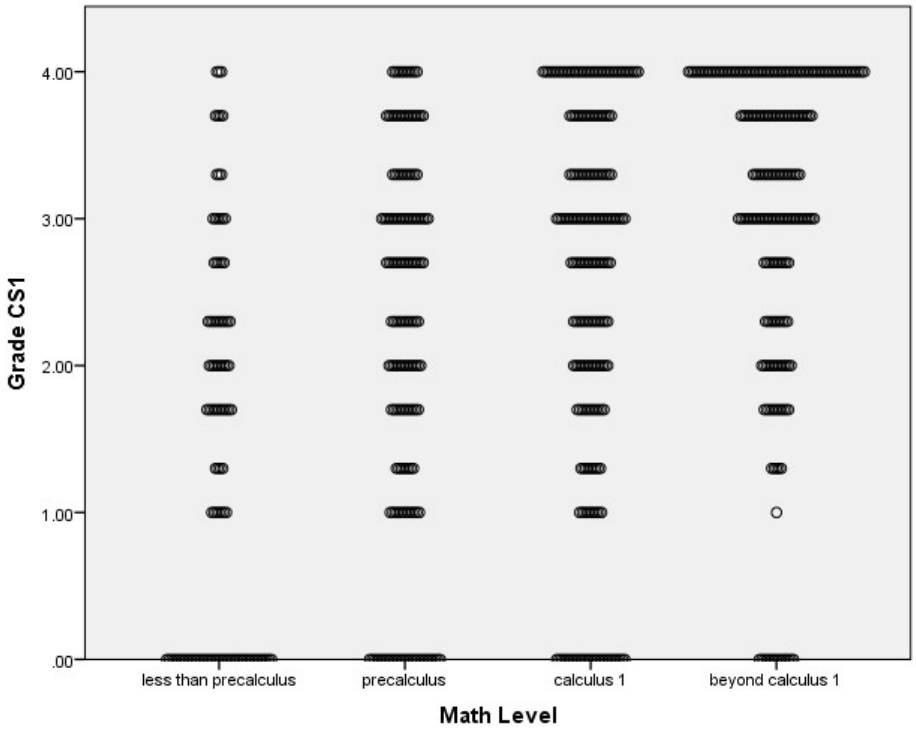


Figure 8: Scatter plot of correlation of math level to CS1 performance

[1] When tested for homogeneity of variance, the variance was found not to be equal (p=.001).

Using the Kruskal-Wallis test (which does not rely on the assumption of normal distribution), the results are the same as the Anova test, with $p<.001$. Using the Mann-Whitney U test to compare math levels two at a time, the results are the same as the Tukey HSD tests, except that for the difference between precalculus and calculus 1, $p=.002$ and for every other difference between math levels $p<.001$. The tests that do not rely on the assumption that the dependent variable is normally distributed thus had even stronger results than those that do rely on that assumption.

The mean of calculus and beyond calculus are both above 2.3, which predicts success in both CS1 and CS2. This measure may be useful despite the lower correlation because it includes students whose math grades are transfers. It does not include those whose most recent math course is more than a year ago because when those students were included, the correlation was low and there was no difference between those who achieved a productive grade in precalculus and those who did not.

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
less than precalculus	106	1.3217	1.29158	.12545	1.0730	1.5704
precalculus	163	2.0988	1.30090	.10189	1.8976	2.3000
calculus 1	212	2.4788	1.31346	.09021	2.3009	2.6566
beyond calculus 1	221	3.0312	1.12858	.07592	2.8816	3.1808
Total	702	2.3897	1.37269	.05181	2.2880	2.4915

Table 5: Mean in CS1 by math level

Performance in math course taken most recently

The correlation between most recent math grade and performance in CS1 is moderate. It is slightly higher for majors other than Computer Science (Table 6).

	Pearson's R	Significance	N
Computer Science	.48	.000	339
Computer Engineering	.52	.000	168
Other	.53	.000	102
All	.50	.000	609

Table 6: Correlation between most recent math course and CS1 by major

Looking at the scatter plot (Figure 9), there does not appear to be any non-linear correlation between performance in the most recent math course and CS1.

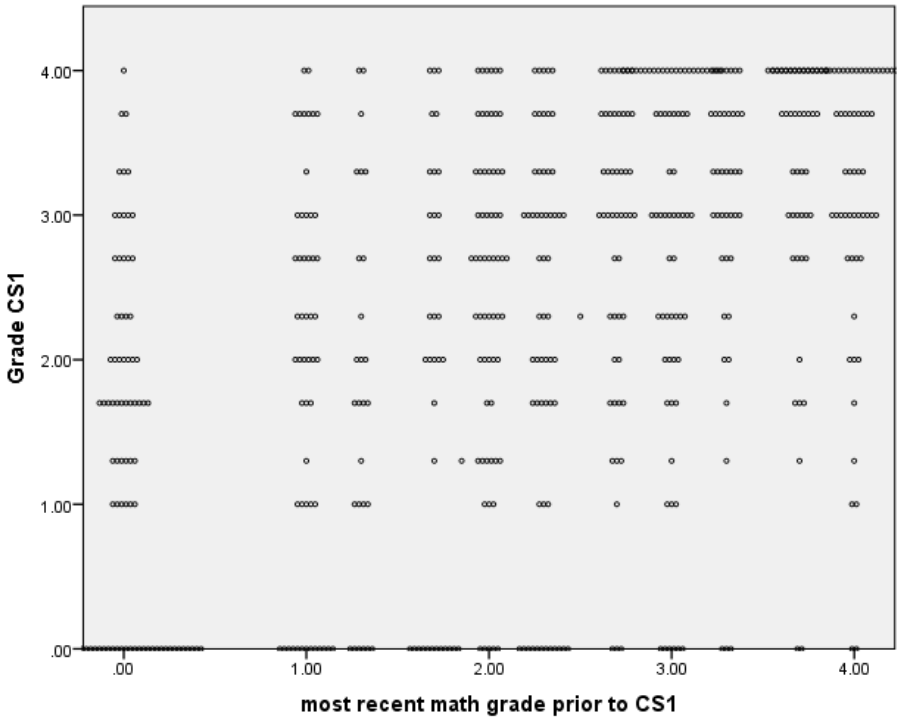


Figure 9: Scatter plot of correlation between most recent math and CS1

Average of math courses taken the previous year

Like the other correlations based on grade, the correlation between recent math average and CS1 grade is moderate (Table 7), though slightly higher than most recent and with a higher N than the individual math courses.

	Pearson's R	Significance	N
Computer Science	.53	.000	348
Computer Engineering	.52	.000	180
Other	.56	.000	104
All	.54	.000	632

Table 7: Correlation between math average and CS1 by major

Again, when we look at the scatter plot of average of math grades from the year prior to CS1 by CS1 grades (Figure 10), the correlation appears to be linear.

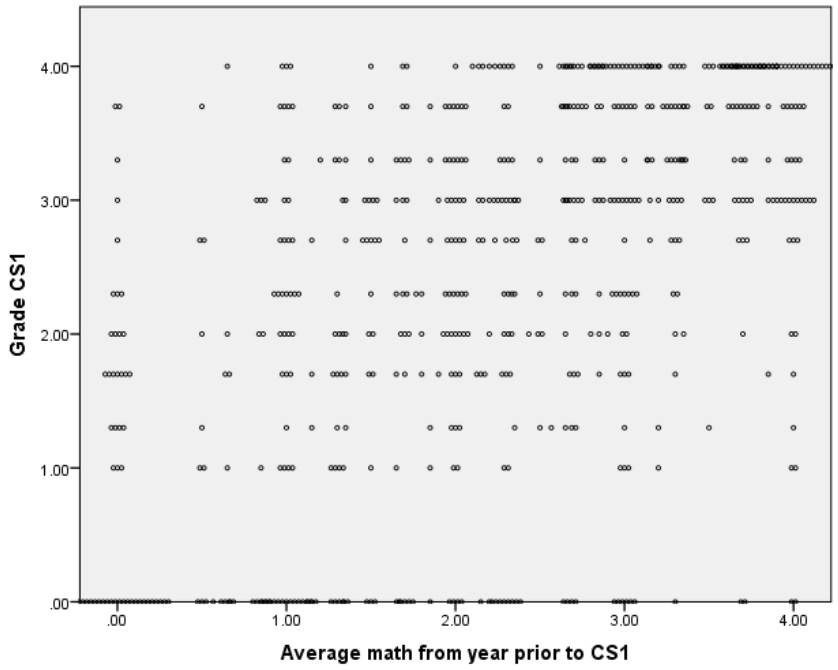


Figure 10: Scatter plot of correlation of average math to CS1

In order to determine if the measure is a good predictor for courses with a more binary distribution, we did a comparison of means based on grade categories derived from the average from the previous year (Table 8 & Figure 11).

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
0	73	1.0438	1.17627	.13767	.7694	1.3183
1	95	1.6958	1.32053	.13548	1.4268	1.9648
2	168	2.2714	1.20479	.09295	2.0879	2.4549
3	172	2.8674	1.12738	.08596	2.6978	3.0371
4	124	3.3831	.93511	.08398	3.2168	3.5493
Total	632	2.4234	1.36288	.05421	2.3170	2.5299

Table 8: Mean in CS1 by math average category

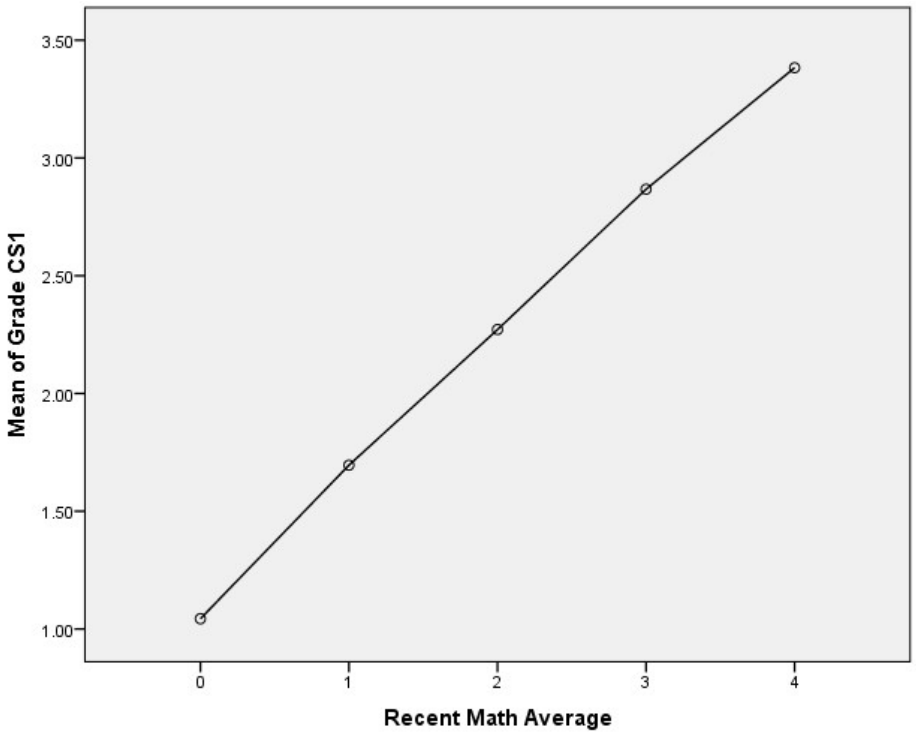


Figure 11: Mean in CS1 grade by math average

The Anova test showed there was a significant effect of math average grades on CS1 grades at the $p<.05$ level for the five grades, $F(4,627)=64.38$, $p<.001$. Post hoc comparisons using the Tukey HSD test show there is a statistically significant difference in the means of performance in CS1 between all of the grade categories.[2] For the difference between F's and D's $p=.003$. For the difference between D's and C's $p=.001$. For the difference between A's and B's $p=.001$. For all other differences between categories $p<.001$.

Using the Kruskal-Wallis test we reject the null hypothesis that distribution of CS1 grades is the same across the categories of recent math average, with $p<.001$. Using the Mann-Whitney U test, there is a difference between every pair of grades. For the difference between F's and D's and the difference between D's and C's, $p=.001$. For the difference between every other pair of grades, $p<.001$. Again, the tests that do not rely on the assumption of normal distribution in the dependent variable corroborate the Anova and Tukey HSD post hoc test results, with even stronger significance.

Predictive power

All of the measures of math achievement based on grade have a moderate correlation, but the one that covers the most students with the highest correlation is the average of the math courses taken in the previous year. The measure based on math level has a lower correlation, but has statistically significant differences between the grades based on level achieved. It has the advantage of including students who have transferred math and those who took precalculus during or after CS1.

[2] Test for homogeneity of variance shows variance is not equal, $p<.001$.

The first question is whether another measure of math achievement could be created based on a combination of the level achieved and the average. In order to do so, the two measures must be independent. When we do a chi-square test of independence, however, we find that math level and math average are definitely not independent $c^2(12, N=632) = 366.9$, $p<.001$. The majority of students who have gone beyond calculus 1 average A's and B's, with almost all of those remaining averaging no lower than a C (Table 9). Those who have not yet passed calculus have a higher percentage of averages in the D and F range. One reason there may be statistically significant differences in mean of grade in CS1 based on math level is that the level may effectively be a proxy for the average grade in math.

	Recent Math Average					Total
Math Level	0	1	2	3	4	
less than precalculus	46	18	0	0	0	64
precalculus	18	38	43	35	27	161
calculus 1	9	32	76	48	29	194
beyond calculus 1	0	7	49	89	68	213
Total	73	95	168	172	124	632

Table 9: Relationship of math level to math average

The next question is whether the average of grades in math courses taken the previous year has predictive power. When we do a linear regression using gender, major, math level and math average as the factors and the grade in CS1 as the dependent variable we find that all these factors account for 31.6% of the variance of the grade in CS1. Average math itself has $R^2=.29$, $F(1, 630) = 256.9$, $p<.001$. The R^2 change when math level is added is .02, $p<.001$. The R^2 change when major is added

is .01, $p<.001$. The three factors that are significant are: math average, $b=.44$, $p<.001$; math level, $b=.18$, $p<.001$; and being a major other than Computer Science or Computer Engineering, $b= -.11$, $p=.002$. Neither the difference between computer science and computer engineering nor gender were significant factors.

The chi-square test showed that math level and math average are not independent. However, the fact that both are significant predictors shows that the level of math achieved does have an impact on the grade. Those who have successfully completed more math do better on average than those who have completed less math.

It is also interesting that being a major other than computer science or computer engineering has a slight negative impact on performance. There were 249 non-majors who took CS1 and they had a higher overall performance than the 683 students majoring in CS or CE. But the linear regression included only students who had taken a math course at this institution during the year prior to CS1. Only 104 (42%) of the non-majors were included, while 528 (77%) of the majors were included. Of the non-majors not included, 14 had transferred a math course, 81 had taken math more than a year prior, 33 were non-matriculating and 17 did not have any math prior.

Without many of the strongest students, the non-majors had slightly worse performance than the majors. The affect of major alone is only 1% since the differences due to math average and math level have already been controlled for. The slightly worse performance is likely due to students taking the course with less prior programming experience. Although majors other than computer science or computer engineering do not need to do as well in CS1 in order to succeed, the higher threshold to identify those likely to struggle may be a good precautionary measure.

The validity of a linear regression relies on four assumptions: that there is a linear correlation between the factors and the dependent variable and insofar as the factors are independent, their influence can be added; that the errors are statistically independent; that the errors have a constant variance; and that the errors are normally distributed. The concern is that the last two assumptions have not been met: the residuals (error terms) do not have a constant variance and they are not normally distributed. For the variance, it is not unusual for the error term to get larger at both ends (Figure 12). It is a bigger issue when they get larger at one end only. The errors are interesting because they show that there are some CS1 grades that are much higher than the model would predict, which is consistent with the unusually large percentage of A's and B's, and there are some CS1 grades that are much lower than the model would predict, which is consistent with the high failure rate in CS1.

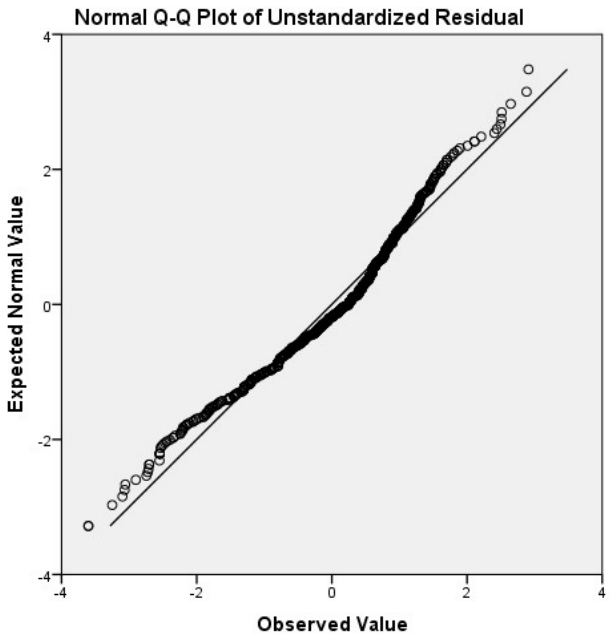


Figure 12: Residual distribution

The concern with the violation of these two assumptions is that it is difficult to calculate the confidence intervals for forecasts. But for the purpose of getting a general idea of how much of the variance is accounted for and seeing which factors are most significant, the violation of these two assumptions does not have any impact, especially since the dataset is large.

Thresholds

The mean of the math average category that is closest to the 2.3 threshold for success is the C-range. Looking more closely at the C's, the divide appears to be between C and C-. Looking at the averages between C and C-, those with a math average of 1.9 or better have a CS1 mean around 2.3, while those below have a CS1 mean closer to 2.0 (or lower). Using the math average as a predictor of success, the threshold would thus be 1.9.

Average math grade may have the highest correlation for the most students, but since the grade distribution in CS1 is not normal, finding thresholds for each math class may be a more useful way to predict success in CS1. Unlike the correlation between the grades in each math class and CS1, to find a threshold we are interested only in the math class taken most recently.[3] The thresholds are based on subjects who took the math course during the year prior to taking CS1.

For precalculus (Table 10), there is a big gap between the mean of those in the C range and those in the B range, and although the difference is not statistically

[3] Since the purpose of looking at the grades in each math class was to see if there is any particular math class that has a higher correspondence, the correlation between grades in each math class and CS1 included the grade if the class were taken in the previous year. This includes grades of students who had taken more math beyond that course prior to taking CS1.

significant, there is not much overlap between the upper bound of the C range and the lower bound of the B range. Those in the C range fall below our threshold for success for majors (2.3), while those in the B range are above. When we look more closely at the B and C precalculus grades (Table 12), we see a small gap between C+ and B- and a much larger gap between B and B+. Those who received a B averaged a grade very close to the threshold for success, while those with a B+ had a mean far above it. The threshold for success for those who have taken precalculus most recently would probably best be placed at B, since the gap between below 2.3 and 2.3 and above falls closest to B.

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
F	19	.8579	1.25756	.28850	.2518	1.4640
D	22	1.4591	1.33440	.28450	.8675	2.0507
C	33	1.7758	1.18058	.20551	1.3571	2.1944
B	40	2.5350	1.21688	.19241	2.1458	2.9242
A	27	2.9852	1.02796	.19783	2.5785	3.3918
Total	141	2.0496	1.37807	.11605	1.8202	2.2791

Table 10: Mean of CS1 grades by precalculus grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
1.70	7	1.7143	1.22533	.46313	.5810	2.8475
2.00	16	1.8250	1.03312	.25828	1.2745	2.3755
2.30	10	1.7400	1.47211	.46552	.6869	2.7931
2.70	16	2.1688	1.32097	.33024	1.4649	2.8726
3.00	12	2.4667	1.39957	.40402	1.5774	3.3559
3.30	12	3.0917	.61120	.17644	2.7033	3.4800
Total	73	2.1918	1.25153	.14648	1.8998	2.4838

Table 11: Means of CS1 grades for precalculus B and C grades

Using the same methodology, the thresholds for applied calculus, introductory calculus and analytic geometry, calculus 2 and linear algebra were determined (Table 12). Since the number taking applied calculus is low, the thresholds were based on all those who had taken it most recently, even if not during the year prior to CS1. For the tables used in determining these thresholds, see Appendix B.

In seeing the wide range of thresholds between the different math courses, it becomes clearer why the correlation between most recent math grade and CS1 grade is moderate and lower than the average of the recent math grades. When a D in calculus 2 corresponds to a CS1 mean that is the same as a B in applied calculus, the grade without the context of the course in which it was earned is only somewhat informative.

There were no grades in any of the math courses that had a mean of 3.7 or higher. While many students earn A's in CS1, no math grade predicts an A.

	Threshold of success (equal or above)
Average of math courses in prior year	1.9
Precalculus	3.0
Applied Calculus	3.0
Intro. Calculus & Analytic Geometry	2.7
Calculus 2	1.3
Linear Algebra	2.3

Table 12: Summary of thresholds

The correlations of individual course grades was used to determine if there was any one course that had a higher correlation, but it included students who had taken courses beyond that course. Revisiting the individual course grades, but restricting it to only those who took that course most recently and calculating the correlations using

the most recent math grade, we find that the correlation changes for precalculus, applied calculus and calc 1.

	Pearson's *R*	Significance	*N*
precalculus	.51	<.001	141
applied calculus 1	.83	.006	9
introductory calculus and analytic geometry	.47	<.001	208
calculus 2	.44	<.001	205
linear algebra	.56	<.001	40
calculus 2 & linear algebra	.85	.033	6

Table 13: Correlation between most recent course grades and CS1

In conclusion, the measure of math achievement with the highest correlation to success in CS1 is the average grade of the math courses taken in the previous year. The fact that this measure alone accounts for almost 30% of the variance in grades indicates that there is, indeed, a deep relationship between math and learning programming. It is also interesting to note that when math courses taken more than a year prior to CS1 are included, the correlation decreases. Only recent math performance has a statistically significant correlation to performance in CS1. Furthermore, which math course was taken most recently affects the threshold that predicts either struggle or likely success in CS1 and CS2.

Research question 2

Looking at the general question whether learning math improves performance in CS1, it is worth drawing attention again to the difference in performance in CS1 between math levels (Table 5). There was a statistically significant difference between

all math levels, with those below precalculus having an average grade of 1.3, those with precalculus having an average grade of 2.1, those with calculus having an average grade of 2.5 and those who had gone beyond calculus having an average grade of 3.0.

While some of the difference in CS1 performance is undoubtedly because the students at higher levels generally had better math grades, this is not the only reason there is a difference in performance between math levels. If it were, then math level would not have been independent enough of math average to be a significant factor in the linear regression. The beta values reflect the amount the grade will change for each unit of change of the factor of which the beta value is the coefficient. The math average has up to four units (0.0 to 4.0), while the math level also has up to four units (-1 to 3). The beta values can be directly compared to see how much impact the math level has independent of math average. For the amount of variance accounted for by math average and math level (30.5%), math level (b=.18) is responsible for ~29%, while math average (b=.44) is responsible for ~71%. The fact that slightly under 9% of the variance is due to math level alone supports the claim that learning math makes a difference.

The first sub-question is whether learning challenging math helps those who struggle with it. Another way of asking this question is whether math prerequisites are helpful or mere gatekeepers. The difference in performance between those who had not yet passed precalculus with at least a C- and those who had, (1.3 versus 2.1) by itself indicates that those who have not yet learned precalculus are not likely to succeed in CS1. Looking more closely at those who had not yet passed precalculus

with at least a C-, the differences become even more striking. There were 99 students who either had taken a math course during the year prior to CS1 or took precalculus for the first time during or after CS1[4]. The 61 students who had either taken calculus and failed or taken precalculus and got below a C- averaged 1.1 in CS1 (median 1.0), while the 38 students who took precalculus for the first time during or after CS1 averaged 1.6 (median 2.0). While the average of both groups is not high, it is important to note that those who cannot do reasonably well in precalculus before they take CS1 are almost guaranteed to struggle in CS1. This suggests that insofar as precalculus is a gatekeeper, it is not an arbitrary one. The students it is holding back are arguably not ready for CS1, and are unlikely to succeed. But this does not answer the questions whether, for those who struggle with it, learning the math improves performance in CS1, or for students who did not struggle with precalculus, whether they would have done just as well in CS1 without having taken it.

Performance improvement from retaking math

The first test looks at whether for someone who struggles with precalculus, retaking it makes a difference in their CS1 grade. The result is the CS1 mean of the group that retook precalculus and got at least a C the second time is between the means of the group that did not learn precalculus and the group that learned it the first time (Table 14). While there is a significant effect of learning precalculus on CS1 grade at the $p<.05$ level, $F(2,137) = 15.72$, $p<.001$, the post hoc Tukey HSD test shows

[4] Table 5 reports the number of students in the below precalculus category as 107. That includes 8 students who had taken their last math course more than a year prior to CS1. Those students were included in the statistics because that category also included students who had no math prior, so for that category the requirement of having taken math recently was not enforced.

that effect is between the group that did not learn precalculus and the group that learned it the first time, $p<.001$. There is no statistically significant difference between the means of those who did not learn precalculus and those who learned it the second time, $p=.264$, or between those who learned precalculus the first time and those who learned it the second time, $p=.059$.

Using the Kruskal-Wallis test, there is a main effect of doing well in precalculus, $p<.001$. The results of the Mann-Whitney U test comparing each pair of groups is different than the results of the Tukey HSD test. There is no difference in grade distribution between those who did not learn precalculus and those who learned it twice, $p=.109$. There is a difference between those who learned precalculus the first time and both those who did not learn precalculus, $p<.001$, and those who learned precalculus the second time, $p=.010$. The conclusion of this test is that if a student does poorly in precalculus the first time, taking it again and doing better does not make any statistically significant difference in performance in CS1. Only earning at least a C- the first time makes a statistically significant difference in grade in CS1.

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
non productive grade(s) only prior	39	1.1974	1.33643	0.21400	0.7642	1.6307
took twice, productive 2nd time	17	1.7647	1.09484	0.26554	1.2018	2.3276
C- or better first time	84	2.5274	1.23086	0.13430	2.2603	2.7945
Total	140	2.0683	1.36483	0.11453	1.8419	2.2947

Table 14: Mean in CS1 by precalculus group

The results for retaking calculus were stronger. While the number of students who retook calculus and raised their grade was slightly less than the number who retook precalculus and raised their grade, the results (Table 15) were different. There was a significant effect of learning calculus on CS1 grade at the $p<.05$ level: $F(2,203)=22.14$, $p<.001$. Post hoc comparisons using the Tukey HSD test show there is a statistically significant difference in grade in CS1 between those who received below a C- in calculus and did not improve their grade prior to taking CS1 and those who earned at least a C- the first time ($p<.001$). There was no statistically significant difference between those who received below a C- and did not improve their grade and those who raised their grade the second time to at least a C ($p =.108$) or between the grades of those who learned calculus the first time and those who learned it the second time ($p = .332$).

The results using the Kruskal-Wallis test corroborated the Anova test showing there is a difference in performance in CS1 between those who learned calculus 1 and those who did not, $p<.001$. The Mann-Whitney U test showed there is a difference in performance between those who did not learn calculus and those who learned it the second time, $p=.036$, and a difference in performance between those who did not learn calculus and those who got at least a C- the first time, $p<.001$. There is no difference between those who learned calculus the first time and those who learned it the second time, $p=.064$.

This supports that, for those who would struggle with calculus, learning calculus does make a difference in the performance in CS1. Although precalculus is the prerequisite for CS1, it appears that, while a necessary condition, it may not be

sufficient. Calculus is the math course with the highest correlation to success in CS1, and succeeding in calculus also helps students to succeed in CS1.

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
less than C-, (if taken twice, less than C 2nd)	71	1.4239	1.25851	.14936	1.1261	1.7218
less than C- first, C or better 2nd	14	2.1714	1.10692	.29584	1.5323	2.8105
C- or better first time	121	2.6769	1.27592	.11599	2.4472	2.9065
Total	206	2.2107	1.38356	.09640	2.0206	2.4007

Table 15: CS1 performance by calculus group

Performance improvement from more math

A paired sampled t-test was conducted to compare performance in CS1 after taking only precalculus and after taking both precalculus and calculus 1, achieving at least a 1.7 in calculus. There was not a significant difference in the scores for precalculus only (M=2.2, SD = 1.31) and precalculus and calculus (M=2.7, SD = 1.01); $t(16)=-1.61$, $p=.127$.

The paired t-test shows that for those with the same grade in precalculus, although the mean is a half grade higher for those who took calculus, because the number of pairs is low, the difference is not statistically significant. Any significant difference in mean between all those taking only precalculus and all those who took precalculus and also calculus is based on those succeeding in calculus having a larger proportion of higher grades in precalculus than those who only took precalculus. The

limitations of the paired t-test are, first that the number of pairs is too low (17) and second, in order to control the test, all pairs took CS1 with the same instructor. The question is whether it is representative of everyone.

In order to see what effect learning calculus has on everyone who took both precalculus and calculus (achieving at least a C-), the CS1 grades are compared to the mean CS1 grade of those who took only precalculus and earned the same precalculus grade (Figure 13). Because comparing a grade to a mean is not informative, the meaningful comparison is using the percentage of students whose grades are more than a grade away from the mean in either direction, comparing those who only took precalculus and those who also passed calculus with at least a C- (Figure 14).

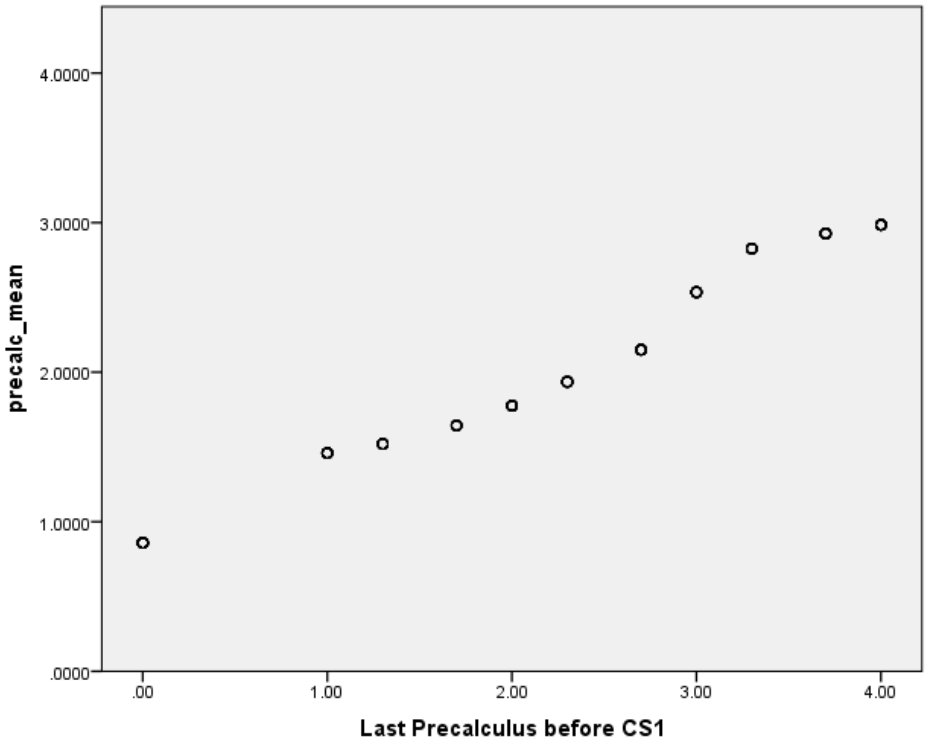
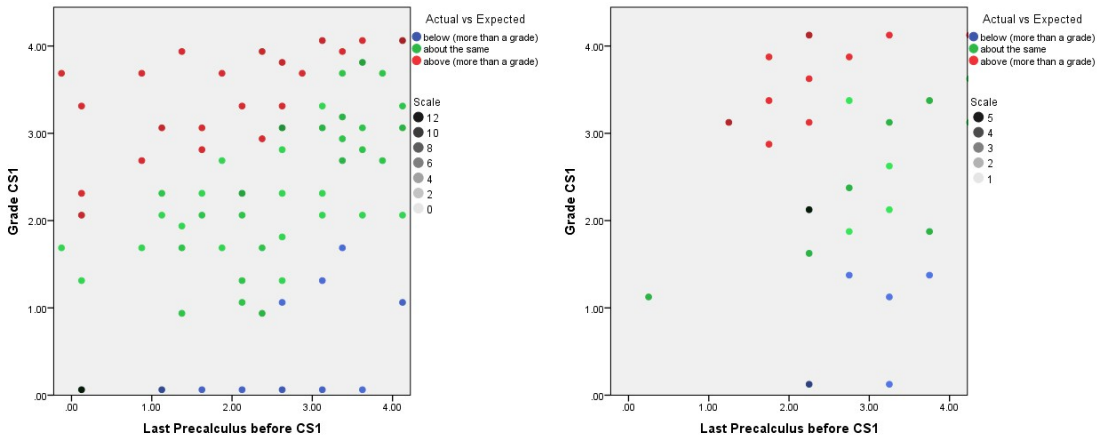


Figure 13: CS1 mean for each precalculus grade



Precalculus only Precalculus & Calculus 1

Figure 14: Precalculus only vs precalculus and calculus

The results are that those with only precalculus (N=141) have 17.7% below, 61.7% the same and 20.6% above. The group that also passed calculus 1 with at least a C- (N=44) has 20.5% below, 52.3% the same and 27.3% above. There is a 2.8% increase in the number whose grade is more than a grade below, and a 6.7% increase in the number whose grade is more than a grade above, for a net increase of 3.9%. A chi-square test shows that there is no statistically significant difference between the proportions in the CS1 grade categories between those who had only precalculus and those who had precalculus and calculus $c^2(2, N=185) = 1.32$, $p=.518$.

We next compared those who have had just calculus 1 (no precalculus) with those who started with calculus 1 and also took calculus 2 or linear algebra. A paired sample t-test was conducted to compare performance in CS1 after taking only calculus 1 and after taking both calculus 1 and calculus 2 or linear algebra, achieving at least a 1.7 in whichever course. There was not a significant difference in the scores for calculus 1 only (M=2.7, SD = .91) and calculus 1 and calculus 2 or linear algebra

(M=3.3, SD = .79); $t(12)= -1.66$, $p=.122$.

When we look at everyone, the results are stronger. Those who have calculus 1 and did not take more math prior to CS1 (N=207) have 24.2% below, 54.1% the same and 21.7% above. Those who also have successfully completed calculus 2 or linear algebra with at least a C- (N=135) have 13.3% below, 60.0% the same and 26.7% above. The net improvement is 15.9%. A chi-square test shows that there is a difference in proportions of CS1 grades between those who have had calculus, but not beyond, and those who have calculus and have also successfully completed calculus 2 or linear algebra $c^2(2, N=342) = 6.15$, $p=.046$.

When we separate those who started with precalculus from those who started with calculus, we see a difference in how much they improve their grade by taking more math. Those who started with precalculus and took calculus 1 (but not further math) prior to CS1 (N=88) have 26.1% below, 55.7% the same and 18.2% above. Those who started with precalculus, took calculus and also successfully completed one of the courses beyond calculus 1 with at least a C- (N=30) have 26.7% below, 53.3% the same and 20.0% above. The net increase is 1.2%. A chi-square test shows that there is no difference in proportions of CS1 grades between those who started with precalculus and took calculus and those who started with precalculus, took calculus and also successfully completed a course beyond calculus $c^2(2, N=118) = .064$, $p=.968$. For those who start with precalculus, learning math beyond calculus results in no performance improvement.

Those who start with calculus and have not taken more math beyond prior to CS1 (N=119) have 22.7% below, 52.9% the same and 24.4% above. Those who start

with calculus and have successfully completed one of the courses beyond calculus 1 with at least a C- (N=105) have 9.5% below, 61.9% the same and 28.6% above. This is a net improvement of 17.4%. A chi-square test shows that there is a difference in proportions of CS1 grades between those who have had only calculus, and those who started with calculus and have also successfully completed calculus 2 or linear algebra $c^2(2, N=224) = 7.01$, $p=.030$. Averaged out, this represents a grade improvement of almost 2 grade increments (e.g. C- to C+).

In conclusion, whether math helps depends on what course a student starts with and how much they struggle. For those who start out in precalculus, the only thing that makes a difference is doing well the first time. For those who do poorly the first time, retaking precalculus and improving their grade does not significantly improve their performance in CS1. Students who cannot get a C- the first time in precalculus are almost guaranteed to struggle in CS1, whereas those who get at least a C- have a

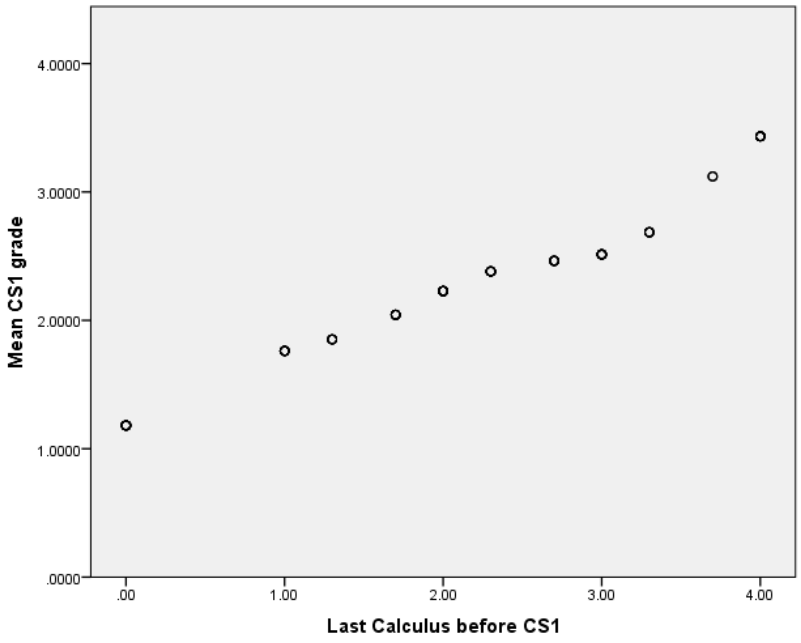
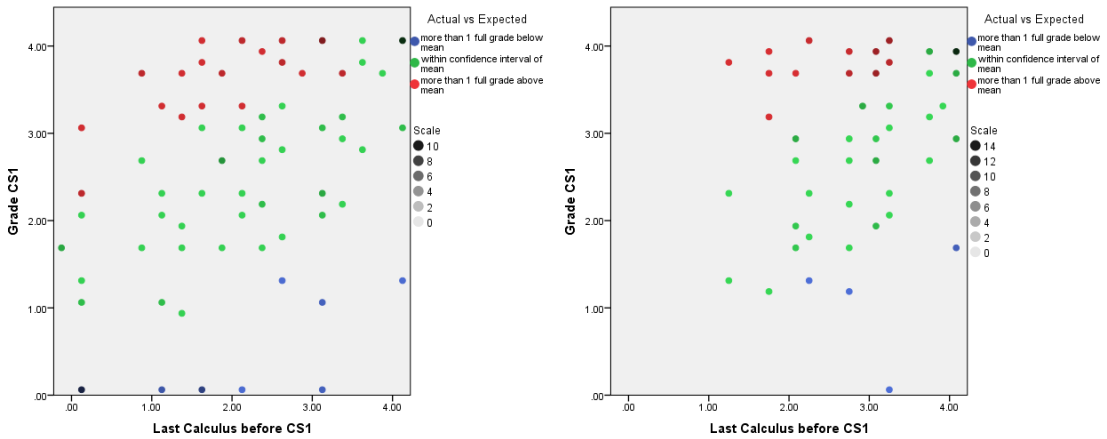


Figure 15: Mean CS1 grade for each calculus 1 grade



Calculus only, no precalculus Calculus & beyond, no precalculus

Figure 16: Calculus only versus calculus and beyond calculus

higher chance at succeeding, and those who get a B or higher are more likely to do reasonably well in both CS1 and CS2. Furthermore, succeeding in courses beyond precalculus does not significantly improve their grade. So for those who start out in precalculus, doing well the first time makes a difference, but neither learning precalculus the second time (for those who didn't do well the first time) nor learning more math (for those who did) appears to have a significant effect in learning computer programming.

For those who start out in calculus it's a different story. For those who struggle, retaking the class and learning the material the second time brings their CS1 grade up to the same range as those who did well in calculus the first time. Not only does learning calculus help those who struggle with it, but for those who start out in calculus, learning math beyond also appears to improve performance.

But here a note of caution is in order. The reason the paired sample t-test showed no statistically significant difference is that there were only 13 pairs. The low number of pairs is not due to a low number of students in either the calculus only or

beyond calculus categories, but because the beyond calculus category was composed predominantly of computer engineering majors, while the calculus only category was dominated by computer science majors. Although the students in the beyond calculus category did better on average than those in the calculus only category who received the same calculus grade as they did, other factors than the following math course may have contributed to their better performance. Computer engineering majors have a different curriculum and, until recently, took CS1 a semester later than the computer science majors. That said, it is worth noting that when comparing students who had the same calculus grade, same major and same gender, the average grade was .6 higher for those who had gone beyond calculus than for those who had only calculus. While the number of pairs was too low to be statistically significant, it does suggest that learning more math may have made a difference.

The divide between starting in precalculus versus starting in calculus, along with the differences between math levels, brings on the question whether those who start out in calculus do significantly better in CS1 than those who start out in precalculus. Those who start out in precalculus and get less than a C-, or who start out in calculus and fail, have a mean grade of 1.2 in CS1 (N=84). Those who start out in calculus and pass with less than a C- have a mean grade of 2.0 (N=69). Those who start out in precalculus and get at least a C- have a mean grade of 2.5 (N=363) and those who start out in calculus and get at least a C- have a mean grade of 2.9 (N=281). There is a statistically significant difference based on starting point, $F(3,793) = 42.64$, $p<.001$. The Tukey post hoc test shows a significant difference between all categories, with the difference between starting out in precalculus with at least a C- and starting

out in calculus with a D having p=.032, while for the differences between all other categories p<.001.

So those who start in calculus and do reasonably well do indeed do better in CS1 than those who are not ready for calculus. That group starts out the highest and benefits the most from learning math. This suggests that while learning math may help them, being strong in math in the first place makes the biggest difference in performance in CS1. That said, those who do respectably in precalculus also generally succeed in CS1, though with a lower average than those who start in calculus. Those who start in calculus and barely pass are more likely to struggle in CS1, while those who cannot pass precalculus with at least a C- the first time (or fail calculus) are much less likely to succeed in learning programming.

Research question 3

Using recent math average, the correlation for women $r(66)$=.59, p<.001 is slightly higher than that of men $r(562)$=.54, p<.001. One of the reasons the correlation overall is moderate is that the grade distribution in CS1 is not normal, with a higher concentration of A's and B's than the math grades are likely to have. Having a higher correlation may indicate fewer A's compared to men.

When, using the grade categories for math average and CS1, we compare the distribution of math grades to CS1 grades for men and women, we find that although 19.5% of the men got A's in math, 29.6% got A's in CS1. More women got A's in math (20.6%), but far fewer got A's in CS1 (22.1%). When we consider A's and B's together, 45.9% of men got an A or B in math, but 55.5% received an A or B in CS1. Although

54.4% of women received an A or B in math, 54.5% received an A or B in CS1.

At the bottom end, far fewer women failed then men. While 10.3% of women failed math, only 5.9% failed CS1. On the other hand 11.7% of men failed math, but 17.2% failed CS1. Taking D's and F's together, 27.3% of men received a D or F in math, and 25% received a D or F in CS1. Among women 20.6% received a D or F in math, but only 16.2% received a D or F in CS1. A similar percentage of women and men received a C in math (26.8% for men, 25% for women), but only 19.5% of men received a C in CS1, whereas 29.4% of women, almost 10% higher, received a C in CS1. This represents a much higher proportion of C's and a much lower proportion of A's compared to the math grades of men. This discrepancy is of concern.

Looking at it further, the real issue is recruitment. Considering all students who took CS1, only 13.3% were women. Only 10.9% of those majoring in Computer Science or Computer Engineering were women. The completion rate of CS2 for women majoring in Computer Science or Computer Engineering was the same as the completion rate for men. This suggests that the percentage of women learning computer programming at this institution is below the national average, which is already alarmingly low.

The difference in CS1 performance between men and women with the same math achievement was due to having most of the small number of women take CS1 with instructors with a closer to normal grading distribution. The high percentage of men throughout the 11 year period meant that their overall grade distributions were affected by all instructors.

What this shows is that a difference in correlation between recent math grades

and CS1 grades between men and women may reveal an underlying discrepancy. In a major that suffers from a huge disparity between men and women, it is important to be aware of any issues, and the correlation of math average to CS1 grades can be used as a tool in this evaluation.

Part 2 data

There were 102 students who volunteered to participate in this study, 59 from the first two semesters and an additional 43 from the third semester. The course assessment between the instructor who taught the first two semesters and the experienced instructor new to teaching CS1 who taught the third semester was determined to be incommensurable. The subjects from the third semester were not included in the study, with the exception of looking at the correlation between math average and SAT scores.

The distribution of CS1 grades of the subjects in the sample is not typical of the grade distribution in CS1. A disproportionate number of students who received A's volunteered for the study while far fewer failing students volunteered. The recruitment for the second semester was done near the beginning of the term before any work was due. It appears that the investment in the course that manifests itself as engagement and better grades is also a significant factor in willingness to participate in a study about the course.

The grade distribution is not normal. Tests for normality, both Shapiro-Wilks and Kolmogorov-Smirnov, pass (N=59, p<.001), so we reject the null hypothesis that there is no difference between this distribution and normal. The Q-Q plot (Figure 17)

indicates a heavier concentration in the left tail (grades of F).

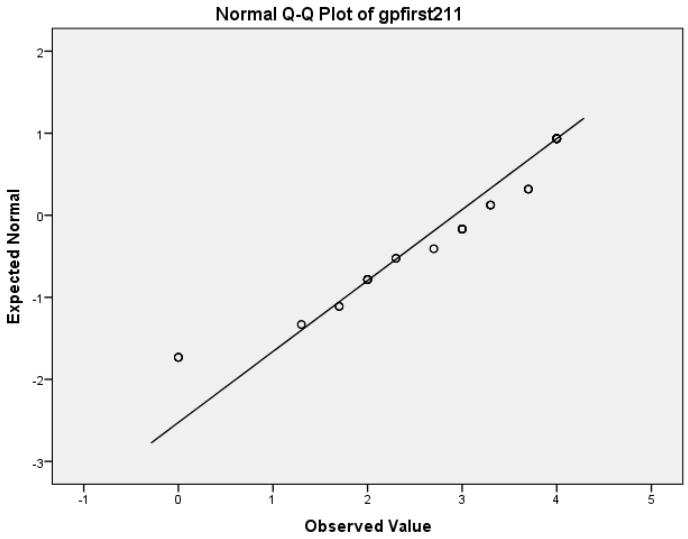


Figure 17: Normality plot of CS1 grades of sample

Regression analysis

Because both math average and math level were statistically significant in the regression analysis of part 1, and also because math average is included in GPA, both math average and math level were included as factors. Only 28 students had all factors. For those students, GPA, verbal and math SAT scores and math average had statistically significant correlations to CS1 grade. For GPA $r(26)=.61$, $p<.001$. For verbal SAT scores $r(26)=.36$, $p=.029$. For math SAT scores $r(26)=.32$, $p=.047$. For average of recent math grades $r(26)=.32$, $p=.048$. For math level $r(26)=.14$, $p=.242$. For computer engineering $r(26)= -.03$, $p=.451$. For other major $r(26)=.03$, $p=.433$. And for gender $r(26)= -.09$, $p=.331$. With all factors, although R^2 is .48, the results are not statistically significant: $F(8,19)=2.20$, $p=.075$. However, removing gender and other major accounts for the same portion of variance and the results are statistically

significant: R^2=.48, F(6,21)=3.25, p=.020. With GPA, verbal SAT scores and math level: R^2=.45, F(3,24)=6.59, p=.002. GPA is the only significant predictor with b=.61, t(24)=3.74, p=.001.

When we remove both math and verbal SAT scores and restrict it to those who have had math in the last year, but do not include math average as a factor, two more subjects are included (N=30) and the amount of variance accounted for increases slightly: R^2=.47, F(2,27)=11.87, p<.001. GPA is the only significant predictor with b=.59, t(27)=3.67, p=.001.

When we remove average math and include SAT scores N=51, but the amount of variance accounted for goes down significantly: R^2=.34, F(4,46)=5.85, p=.001. Neither SAT math scores nor math level makes a difference. With only GPA and SAT verbal scores, R^2=.34, F(2,48)=12.13, p<.001. GPA alone accounts for most of the variance: R^2=.23, F(1,49)=14.51, p<.001. When SAT verbal scores are added, the R^2 change=.11, F change = 7.75, p=.008. GPA is the more significant predictor with b=.38, t(48)=3.07, p=.003, but the SAT verbal score is also a predictor with b=.34, t(48)=2.78, p=.008.

The residual distribution for the linear regression with 28 subjects is not different than normal according to both the Shapiro-Wilks (p=.050) and Kolmogorov-Smirnov (p=.200) tests. The same is true for the residual distribution for the linear regression with 30 subjects (in Shapiro-Wilks p=.429 and in Kolmogorov-Smirnov p=.200). The residual distribution for the linear regression with 51 subjects passes the Shapiro-Wilks test, p=.013, so it is different than normal. While the formula for the linear regression that does not include math average may not be accurate for

forecasting a grade from the predictors, the variance it accounts for and the predictors it identifies are still valid.

While math average was not a significant predictor even in the two models which included it as a factor, GPA includes those math courses. The fact that when math average is included, the amount of variance accounted for is significantly higher, indicates that GPA is a better predictor when the student has had math recently. When the student does not have any recent math courses, using a combination of GPA and SAT verbal scores is an alternative for predicting performance in CS1, though its predictive power is less than when the student has taken math courses recently.

It is interesting that SAT math scores are not a significant factor. Using all of the students (including those in the third semester), there are 55 students who have both recent math and SAT scores. For that set, there is a moderately low correlation between SAT math and average math grades: $r(53)=.39$, $p=.003$. Between the fact that SAT math scores are not a significant factor and the fairly low correlation with math average, they probably are not a good substitute when math average is missing.

Summary of results

Doing well in math predicts success in learning programming. Those who struggle with precalculus, or who jump ahead into calculus and fail, are far less likely to succeed. Those who struggle and retake precalculus and learn the material the second time may raise their CS1 grade somewhat, but not enough to be statistically different than the grade they would have received had they not retaken precalculus, nor enough to predict success. Those who get poor grades in calculus 1 will benefit

from retaking calculus and learning the material. Retaking calculus and earning at least a C does predict success in CS1.

Those who have done well in math courses after calculus tend to do the best in CS1. Those who start in calculus not only benefit from retaking calculus if they do poorly, but they also benefit from learning more math beyond calculus. Those who start in precalculus do not improve their grades significantly by learning calculus. The starting point does make a significant difference in overall performance. This suggests that strong math aptitude, as reflected in starting with calculus and doing well in it, may have more to do with success in learning programming than learning more math. There is significant improvement for those who start out in calculus and succeed in calculus 2 and/or linear algebra, but the difference in grade may also be due to the difference between the curricula of computer engineering versus computer science, since the majority of the students who have gone beyond calculus 1 before taking CS1 are computer engineering majors.

The strongest predictor of success is GPA at the time of taking CS1, as long as that GPA includes recent math grades. The average of recent math grades is also a predictor, and the threshold below which students are likely to struggle in CS1 is 1.9. Alternatively, the grade in the most recent math course can be compared to the threshold for success for that math course (see Table 12).

The correlation between average math grade and CS1 grade can also be used as an instrument to detect discrepancies in performance in CS1 between men and women with the same math ability. Even a relatively small difference may indicate an issue that can be further scrutinized by comparing the categories of grades corresponding to

the math average to the categories of grades for CS1.

Chapter 5

Conclusions and recommendations

Introduction

The motivation behind this research was to help those students whose primary challenge in learning programming was not “getting it.” Previous research showed that a large portion of success has to do with having versus lacking interest in programming, whether students are motivated to put in extra effort or want to skate by with the bare minimum, or other issues having more to do with attitude. I saw too many students who were interested and willing to do the work, but still had a hard time understanding object-oriented programming. I was also concerned that some of these students may have been coming from backgrounds which did not adequately prepare them, and suspected that math was the key to identifying both those likely to struggle and the nature of the underlying problem.

Many students who struggle the first time take the course again. I have had several students who were excellent the second time because something “clicked” for them, and now it all makes perfect sense. While I applaud these students for sticking with it, I wondered whether there was a way of identifying them ahead of time and offering them more help originally, so they did not need to do poorly and then have to retake the course.

The previous research most relevant to this study is Barker and Unger's instrument based on Piaget's levels of intellectual development. If someone has a difficult time thinking abstractly, they are going to find a programming paradigm

based on defining types and using instances of a type (objects) to be very challenging, perhaps impossible, to grasp. Barker and Unger's diagnostic tool used math and logic to identify who was in the late concrete, early formal and late formal level. This connection of success in math to success in learning programming would be accounted for by the ability to think abstractly that underlies both.

Another issue with math and programming is the question of math prerequisites for programming classes. Are they justified? Do they help students succeed or are they merely a barrier to entry? If the students are not going to be using in programming the math they are required to take, should the prerequisite instead be something that is directly useful in learning programming? Does a math prerequisite whose techniques are not used have value?

This study, thus, set out to further explore the nature of the connection between math and learning programming. Can we narrow down what it is about math that helps? Can we use it as a predictor of who is most likely to struggle? Does math ability predict who succeeds in computer science? Can it be used to detect inequity in the classroom?

Conclusions

The problem that motivated this study is the high failure rate in CS1. The first significant finding is that the grade distribution is heavy on both ends and light in the middle. This suggests that, unlike most courses where grade is largely the result of the application of the student, when learning programming students either get it or they do not. This is not an idiosyncrasy of one instructor, and given the worldwide problem of

the high failure rate, is likely to be true for CS1 courses everywhere. Furthermore, the grades are consistent with performance in CS2. This means that "getting it" in CS1 is essential for success in later courses.

The second significant finding is that math does indeed play a role in whether a student "gets it" or not. Approximately 31% of the variance in grade is based on how well one does in math and what level one has achieved. Doing well is most important, but the math level accounts for almost 9% of the variance in the grade.

At the low end, those who take calculus before they are ready and fail, or take precalculus and get less than a C- are very unlikely to succeed in learning programming (the first time). Their average grade is 1.1, which not only reflects not having mastered the material in CS1, but also predicts not passing CS2. Unfortunately, those who struggle this much with math are not able to improve their CS1 grade significantly by retaking precalculus and improving their grade the second time.

Those who start out in precalculus and get at least a C- the first time do significantly better than those who struggle in precalculus. Their average grade of 2.5 is over the 2.3 threshold for success. Taking more math does not significantly improve their performance.

Those who start out in calculus have three advantages. First, if they do worse than 1.7 the first time, if they retake calculus and learn it the second time, their grade in CS1 improves to the point that it is not statistically different than that of those who got at least a C- the first time. Second, the average CS1 grade for those who get at least a 1.7 the first time is 2.7, which is above the threshold that predicts success. Third, should they take more math, their performance in CS1 is likely to improve

further. Those who also succeed in calculus 2 and/or linear algebra average 3.0 in CS1.

While starting point and math level do make a difference, the recent math average is most critical. While the correlation between math average and CS1 grade is moderate, the mean CS1 grade for each grade category (A, B, C...) for the math average is almost a straight line, from those failing math getting a 1.0 up to those averaging an A in math getting a 3.4. Doing well in math regardless of math level predicts strong CS1 performance. This is why the thresholds for success are based either on recent math average or on the grade in the math course taken most recently rather than on math level.

As long as the student has taken math within the previous year, their GPA can be used to predict how well they will perform in CS1. By itself GPA accounted for 47% of the variance in the grade, but it should be noted that the subjects used in this linear regression almost all had grades C or above. It is not clear whether GPA would account for the same proportion of the variance with a more typical grade distribution. When students had not taken math in the previous year the predictive power of GPA went down considerably. GPA and SAT verbal scores together account for 34% of the variance.

Not succeeding in precalculus predicts struggle, while achieving at least a B predicts that the student is likely to succeed in both CS1 and CS2. This indicates that learning precalculus well does improve performance in CS1. Currently the prerequisite is passing precalculus. If a minimum of C- were required, the math would be a further barrier to success. On the one hand, those who would be affected are those most likely to struggle. On the other, if remediation is successful in helping struggling students to

grasp the concepts, it might be better to identify those students who do worse than a C- rather than bar them from taking the class until they improve their precalculus grade.

We know from the consistency of the CS1 grades with the CS2 grades that the high concentration of A's is not a result of grade inflation. The fact that students with lower math averages can do well in CS1 shows that even those who struggle can succeed if given proper support. Using the math average as a diagnostic tool, or, using the most recent math grade compared to the threshold for the math class taken can help identify those students likely to struggle before the semester begins.

Math average can also be used as a tool to detect whether women and men with the same math achievement succeed at the same rate in learning programming.

Limitations

This study looks at the role of math, but does not include logical aptitude as a predictor. This is because I could find no validated instrument measuring logical aptitude. Students are not required to take logic prior to taking CS1. Coming from a background of philosophy myself, I strongly suspect that logic is even more important than math, and is probably more directly relevant to success in programming. But without any way to measure logical aptitude or achievement, I could not include it in the model.

The other limitation is that, even when the instructor is not present in the room, the students who are less invested in the course are far less likely to volunteer for the study. The small size of the sample and the fact that not all of them had SAT scores and at least one recent math class meant that the results are based on 28-30 students,

almost all of whom got a C or better. While GPA is the most significant predictor, it should be noted that this is true only for those who have had recent math. Since the results from part 1 were based on a much larger sample, the average math grade may be better for the entire population, especially those with a low math average.

Discussion and recommendations

The much higher percent of A's in CS1 compared to math grades shows that those who may be predicted to struggle can still succeed. Looking at the most vulnerable population, those who struggle with precalculus, even though they are far more likely to do poorly in CS1, they are not doomed to fail. Looking at the computer science and computer engineering students who struggled with precalculus and took CS1 prior to 2015, while almost two-thirds either failed CS1, did not continue to CS2 or failed CS2, the other third were able to complete both CS1 and CS2, and one eighth did so with A's or B's in both (Figure 18). The point of prediction is not to deny entry, but to provide remediation. The next question is what sort of remediation would be appropriate.

The first recommendation is to have an extra class section every week and make that mandatory for those who are likely to struggle. Use that extra time to have students carefully walk through what is going on under the hood. If the difficulty is in grasping abstract models, make those models more concrete by having them write down on paper exactly what is going on in memory when two references are declared and objects are instantiated. The more they can see what is going on, the better they will understand the underlying model. In my last two semesters teaching CS1 I

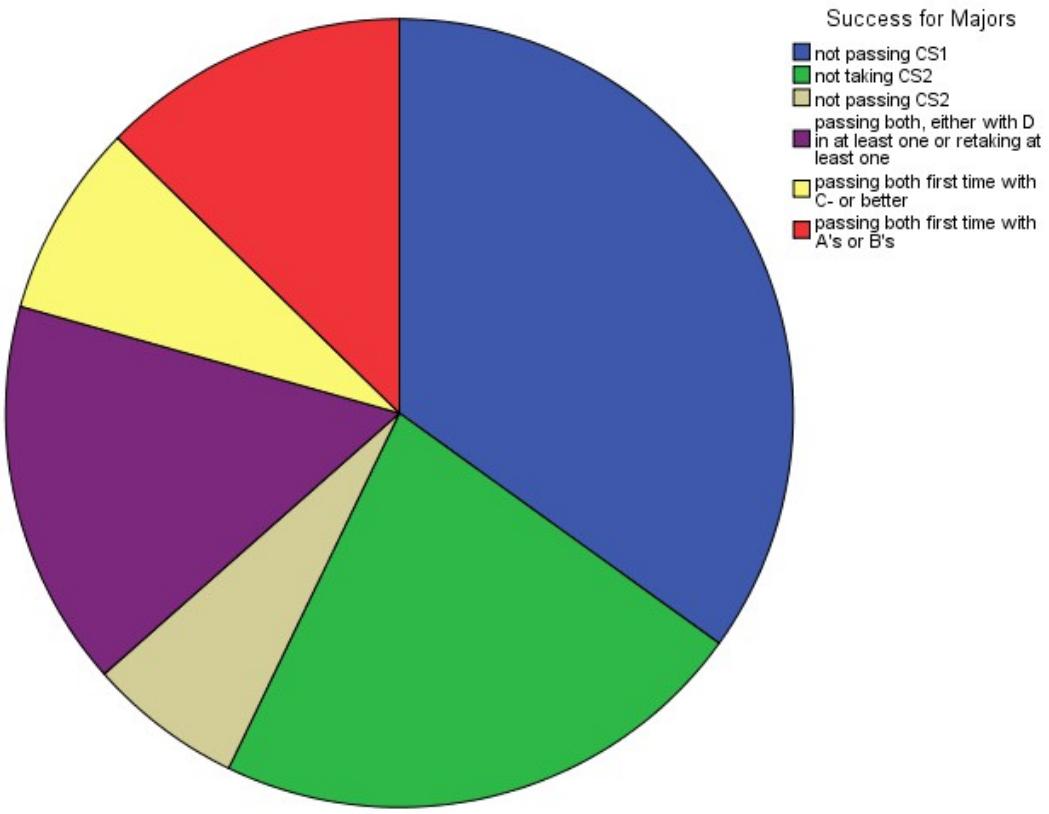


Figure 18: Success rate for CS & CE majors who struggled in precalculus

devoted the last 20-25 minutes of class to group work, including working on exercises like these. The time allotted was sufficient for the better students, but often the weaker students would have benefited from spending more time.

Using math grades as a diagnostic tool may show who is likely to struggle, but the reason they are struggling may not be the same. If someone has not been able to pass precalculus, then their issue is more likely to be abstract thinking. If someone did fine in precalculus and then did far worse in calculus 1, it seems unlikely that their ability to think abstractly is the problem. The fact that there is still a correlation between poor performance in math and poor performance in CS1 shows that there is a

problem, but it may have more to do with life issues, lost confidence or attitude problems than with not being able to grasp the concepts. Nonetheless, the structure of the extra class period may still be helpful. If a student is feeling overwhelmed, giving him/her a task that seems easier may restore lost confidence.

The next steps for this research would be twofold. One question is how effective the above remediation is. To test that would require implementing the extra class time and then seeing whether the mean CS1 grade of the students attending is higher than that of those from the previous semesters with the same math average. Another question is how much of a role does logic play. To test that would require having a group of students take logic prior to CS1 and do a regression analysis using both their grades in logic and their math grades to see how much of a factor their logic grade is in their success.

The question has been asked whether another math course besides precalculus would do as well as a prerequisite for the course. The results of this study show that passing precalculus the first time with at least a C- makes a difference, but this does not answer the question whether another math course could do just as well or better. What it does say is that struggling with precalculus predicts struggling with CS1. If a substitute math course is successful in better preparing students for CS1, the students coming out of that course should also have an easier time with precalculus.

Some of the motivation for finding an alternative is to find a rigorous math course that has more to do with the problems that students in CS need to solve, like discrete math (which is required for Bachelor of Science students later in the curriculum). If such a math course is available, it would be interesting to test how

students taking that course do compared with precalculus. But if the motivation for finding an alternative is that precalculus is seen as a barrier to entry, the results of this research show that succeeding in precalculus the first time correlates with students succeeding in learning programming, and that those who have not achieved at least a C- in precalculus prior to CS1 are far more likely to be over their head in CS1. At the very least taking precalculus can make it possible to predict that the student will struggle so they can receive the extra help they need.

On the one hand, the results indicate that those who are good at math: those who can jump into calculus 1 or even calculus 2 and get consistently good grades, will find learning programming far easier than those who struggle with math. The grade distribution that is heavy on both ends and light in the middle is a product of getting it or not getting it. On the other hand, those who struggle can still succeed. Although only half the students passed both CS1 and CS2 with a C- or better the first time, another quarter did pass both courses. And the error rate for prediction is high precisely because students whose math grades predict doing poorly in CS1 can still do well, given sufficient support.

Appendix A

Explanation of Statistical Terminology

b: The beta values are the coefficients of the independent variables in the regression equation. Each beta represents how much the dependent variable will change for each unit of change of that independent variable. The larger beta is, the more significant that factor is.

p: Also known as significance, this is the probability that the result would fall within a confidence interval, usually 95%. The smaller the *p* value, the less likely that the result is within expected values. If it is less than the confidence interval (usually .05), we reject the null hypothesis that there is no difference. This indicates that the difference is statistically significant.

r: Pearson's r is the correlation coefficient for linear correlations. It is between -1 and 1. When it's negative, the correlation is inverse. The larger the magnitude, the stronger the correlation. The closer it is to 0, the weaker. A correlation with magnitude .5 is moderate.

R^2: The square of Pearson's *r* is used to measure the amount of variance accounted for by factors in a linear regression.

Appendix B

Tables used in finding thresholds

The following tables were used to determine the math threshold below which a student is likely to struggle in CS1. They are based on students who took this math course most recently prior to CS1. With the exception of applied calculus, the students took this math course during the year prior to CS1.

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
F	19	.8579	1.25756	.28850	.2518	1.4640
D	22	1.4591	1.33440	.28450	.8675	2.0507
C	33	1.7758	1.18058	.20551	1.3571	2.1944
B	40	2.5350	1.21688	.19241	2.1458	2.9242
A	27	2.9852	1.02796	.19783	2.5785	3.3918
Total	141	2.0496	1.37807	.11605	1.8202	2.2791

Table 10(repeated): Mean of CS1 grades by precalculus grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
1.70	7	1.7143	1.22533	.46313	.5810	2.8475
2.00	16	1.8250	1.03312	.25828	1.2745	2.3755
2.30	10	1.7400	1.47211	.46552	.6869	2.7931
2.70	16	2.1688	1.32097	.33024	1.4649	2.8726
3.00	12	2.4667	1.39957	.40402	1.5774	3.3559
3.30	12	3.0917	.61120	.17644	2.7033	3.4800
Total	73	2.1918	1.25153	.14648	1.8998	2.4838

Table 11 (repeated): Means of CS1 grades for precalculus B and C grades

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
D	1	.0000	.	.	.	.
C	5	1.6000	1.09316	.48888	.2427	2.9573
B	9	2.1111	1.50204	.50068	.9565	3.2657
A	8	3.0375	.97385	.34431	2.2233	3.8517
Total	23	2.2304	1.37789	.28731	1.6346	2.83

Table 16: Mean of CS1 grades by applied calculus grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
2.70	3	1.6667	1.45717	.84130	-1.9531	5.2865
3.00	4	2.2500	2.06155	1.03078	-1.0304	5.5304
3.30	2	2.5000	.28284	.20000	-.0412	5.0412
Total	9	2.1111	1.50204	.50068	.9565	3.2657

Table 17: Mean of CS1 grades for applied calculus B grades

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
F	43	1.1814	1.07576	.16405	.8503	1.5125
D	33	1.8909	1.31181	.22836	1.4258	2.3561
C	65	2.1215	1.37301	.17030	1.7813	2.4618
B	52	2.6712	1.19169	.16526	2.3394	3.0029
A	24	3.4333	.81862	.16710	3.0877	3.7790
Total	217	2.1770	1.37481	.09333	1.9930	2.3609

Table 18: Mean of CS1 grades by calculus 1 grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
1.70	17	1.7765	1.51432	.36728	.9979	2.5551
2.00	24	2.5042	1.16189	.23717	2.0135	2.9948
2.30	24	1.9833	1.42818	.29153	1.3803	2.5864
2.70	11	2.9818	.99581	.30025	2.3128	3.6508
3.00	27	2.5889	1.35088	.25998	2.0545	3.1233
3.30	14	2.5857	1.02721	.27453	1.9926	3.1788
Total	117	2.3658	1.31897	.12194	2.1243	2.6073

Table 19: Mean of CS1 grades by calculus 1 B & C grades

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
F	21	1.6190	1.38695	.30266	.9877	2.2504
D	22	2.2636	1.19026	.25376	1.7359	2.7914
C	52	2.6385	1.17609	.16309	2.3110	2.9659
B	57	3.2175	1.03236	.13674	2.9436	3.4915
A	55	3.4055	.98778	.13319	3.1384	3.6725
Total	207	2.8585	1.24148	.08629	2.6883	3.0286

Table 20: Mean of CS1 grades by calculus 2 grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
1.00	16	2.2188	1.31819	.32955	1.5163	2.9212
1.30	6	2.3833	.84479	.34488	1.4968	3.2699
1.70	11	2.2182	1.56321	.47133	1.1680	3.2684
2.00	21	2.9333	.94833	.20694	2.5017	3.3650
2.30	20	2.5600	1.12923	.25250	2.0315	3.0885
Total	74	2.5270	1.18475	.13772	2.2525	2.8015

Table 21: Mean of CS1 grades by calculus 2 D & C grades

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
F	5	1.2800	1.22352	.54717	-.2392	2.7992
D	3	2.1333	1.91398	1.10504	-2.6213	6.8879
C	6	2.0000	1.35351	.55257	.5796	3.4204
B	15	3.2533	.82624	.21333	2.7958	3.7109
A	12	3.3333	1.13805	.32853	2.6103	4.0564
Total	41	2.7707	1.31781	.20581	2.3548	3.1867

Table 22: Mean of CS1 grades by linear algebra grade

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean	
					Lower Bound	Upper Bound
1.70	2	.8500	1.20208	.85000	-9.9503	11.6503
2.00	2	1.6500	.49497	.35000	-2.7972	6.0972
2.30	2	3.5000	.28284	.20000	.9588	6.0412
2.70	6	3.0000	.86718	.35402	2.0900	3.9100
3.00	6	3.1833	.90203	.36825	2.2367	4.1300
3.30	3	3.9000	.17321	.10000	3.4697	4.3303
Total	21	2.8952	1.12804	.24616	2.3818	3.4087

Table 23: Mean of CS1 grades by linear algebra B & C grades

Bibliography

Barker, R. & Unger, E. A.. (1983). A predictor for success in an introductory programming class based upon abstract reasoning development. *SIGCSE '83 Proceedings of the fourteenth SIGCSE technical symposium on Computer science education,* 154-158.

Bennedsen, J. & Caspersen, M. (2007). Failure rate in introductory programming. *ACM SIGCSE Bulletin* , *39*(2), 32-36.

Cheryan, S., Master, A., & Meltzoff, A. (2015). Cultural stereotypes as gatekeepers: increasing girls' interest in computer science and engineering by diversifying stereotypes. *Frontiers in Psychology, 6*, 49-57.

Cohen, C. & Deterding, N. (2009). Widening the Net: National Estimates of Gender Disparities in Engineering. *Journal of Engineering Education, 98*(3), 211-226.

Hacker, A. (2012, July 29). Is Algebra Necessary. *New York Times*, p. SR1. Retrieved March 30, 2017, from http://www.nytimes.com/2012/07/29/opinion/sunday/is-algebra-necessary.html

Leeper, R. R. & Silver, J. L. (1982). Predicting Success in a First Programming Course. *SIGCSE '82 Proceedings of the thirteenth SIGCSE technical symposium on Computer science education*, 147-150.

Marion, W. (1999). CS1: What should we be teaching? *ACM SIGCSE Bulletin* , *31*(4), 35-38.

National Center for Education Statistics, (2015), *Bachelor's degrees conferred by postsecondary institutions, by race/ethnicity and field of study: 2012-13 and 2013-14* [Data file]. Retrieved September 29, 2016 from http://nces.ed.gov/programs/digest/2015menu_tables.asp.

PayScale Human Capital, (2016), *2016-2017 PayScale College Salary Report*. Retrieved September 30, 2016, from http://www.payscale.com/college-salary-report/majors-that-pay-you-back/bachelors.

Sheard, J. & Hagan, D. (1998). Our failing students: A study of a repeat group. *ITiCSE '98 Proceedings of the 6th annual conference on the teaching of computing and the 3rd annual conference on Integrating technology into computer science education: Changing the delivery of computer science education*, 223-227.

Simon, Fincher, et al. (2006). Predictors of success in a first programming course. *ACE '06 Proceedings of the 8th Australasian Conference on Computing Education, 52*, 189-196.

Validity Studies. (2017). Retrieved March 29, 2017, from http://research.collegeboard.org/programs/sat/data/validity-studies.

Watson, C. & Li, F. (2014). Failure rates in introductory programming revisited. *ITiCSE '14 Proceedings of the 2014 conference on Innovation & technology in computer science education*, 39-44.

Werner, L., Hanks, B., & McDowell, C. (2004). Pair-Programming Helps Female Computer Science Students. *ACM Journal of Educational Resources in Computing, 4*(1), 1-8.

Wilson, B. & Shrock S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. *SIGCSE '01 Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*, 184-188.