5-2013

A Qualitative and Evaluative Study on Recruiting and Retaining Students in College Computer Science Programs

Matthew Gardner
Utah State University

A QUALITATIVE AND EVALUATIVE STUDY ON RECRUITING AND RETAINING STUDENTS IN COLLEGE COMPUTER SCIENCE PROGRAMS

by

Matthew Gardner

A thesis submitted in partial fulfillment of
the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

Dr. Vicki H. Allan Major Professor	Dr. Xiaojun Qi Committee Member
Linda P. DuHadway Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah 2013

The National Science Foundation through grant CPATH: Computational Thinking Showcase: Computing Concepts Across the Curriculum (D: 0829563) provided financial support for this project.

ABSTRACT

A Qualitative and Evaluative Study on Recruiting and Retaining Students in College Computer Science Programs
by

Matthew Gardner, Master of Science

Utah State University, 2013

Major Professor: Dr. Vicki Allan

Department: Computer Science

This thesis discusses the retention techniques that were learned in the surveys of our study, which include the senior students who were close to graduating from computer science, professionals who are working in a field related to computer science, and first semester computer science students.

This thesis discusses observations from the survey including seniors' idea to create a pre-computer science course for beginners in order to improve retention of those with less experience coming into the major. This thesis also takes into account the least desirable things about the major and ideas to increase the number participating in the major. We discuss reasons the students choose against joining the major in the first place and the seniors' ideas to address these reasons.

We mention the best part of a computer science career is that it keeps professionals learning and interested. We discuss whether this is something that is possible to teach or sell to earlier students to get them interested in the major. We also discuss the possibility of giving more exposure about the major to high-school age students and some possibilities to overcome the stigma generally attached to computer science professionals.

(140 pages)

PUBLIC ABSTRACT

A qualitative and evaluative study on recruiting and retaining students in college computer science programs

Matthew A. Gardner

Computer science is a discipline that is increasing in importance and value in our society, yet we are still failing to graduate a sufficient number of students to keep up with the demand required in the United States economy. We research several ways to retain students. We also discuss ways to increase students' interest in the major, i.e., those who normally would not know about computer science. We discuss ways to increase female participation as well as overall participation in the major.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Computer Science needs more graduates. Universities in the United States are not graduating a sufficient number of computer scientists to keep up with demand in the marketplace. This thesis begins with details of the need for more graduates. Our study focuses on solutions to this problem. We present recommendations based on the data we have gathered.

We have split the results section of this thesis into two major sections: Recruitment and Retention. Both of these play a crucial role in increasing the number of graduates. A greater number of graduates will result from coupling recruitment and retention.

CHAPTER 2

BACKGROUND

Most students pursuing a college degree expect to prepare for a career related to their major of study. Zhang shows that one primary reason students choose a major is because of the long-term financial reward [1]. We want to get those into the major who would be successful and complete the major; likely going on to a successful career in their chosen emphasis. We do not want to attract those who do not have the aptitude to succeed.

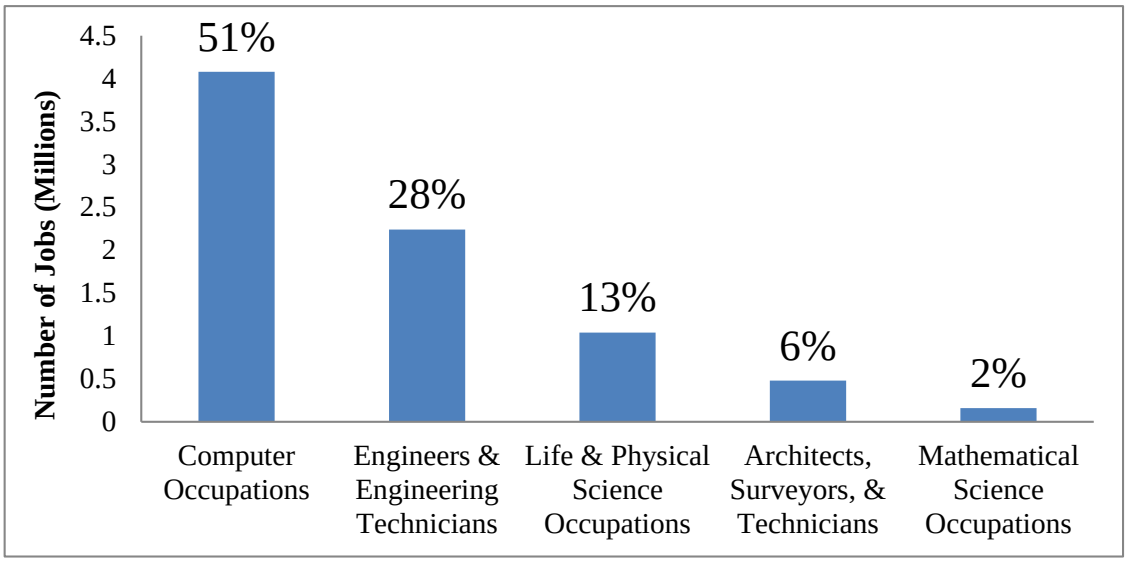

Figure 2.1: Computer occupations dominate STEM: 2018. Source Georgetown Center on Education and the Workforce, STEM. Used with permission.

According to our survey, 45% of seniors in computer science chose the major because of career benefits. This suggests that rewards associated with a computer science major are great incentives to get students into the major. Computer science

(with the exception of the dot-com bust in the early 2000's) has consistently out-performed many other disciplines in career opportunities and salary and is projected to out-perform all other Science, Technology, Engineering and Math (STEM) disciplines through 2018 [2]. Figure 2.1 shows these projected numbers.

If we were to only take into account the correlation between jobs available and enrollment in the major, we would expect to see computer science higher in enrollment than other STEM disciplines. While there has been a recent upsurge in enrollment in computer science [3], there is also projected to be a far more jobs available than can be filled with the current pipeline coming out of college. According to the most recent data reported from the computing research association and the bureau of labor statistics, there will be more than twice the number of jobs available for students graduating in computer science than there will be graduates to fill them (see Figure 2.2).

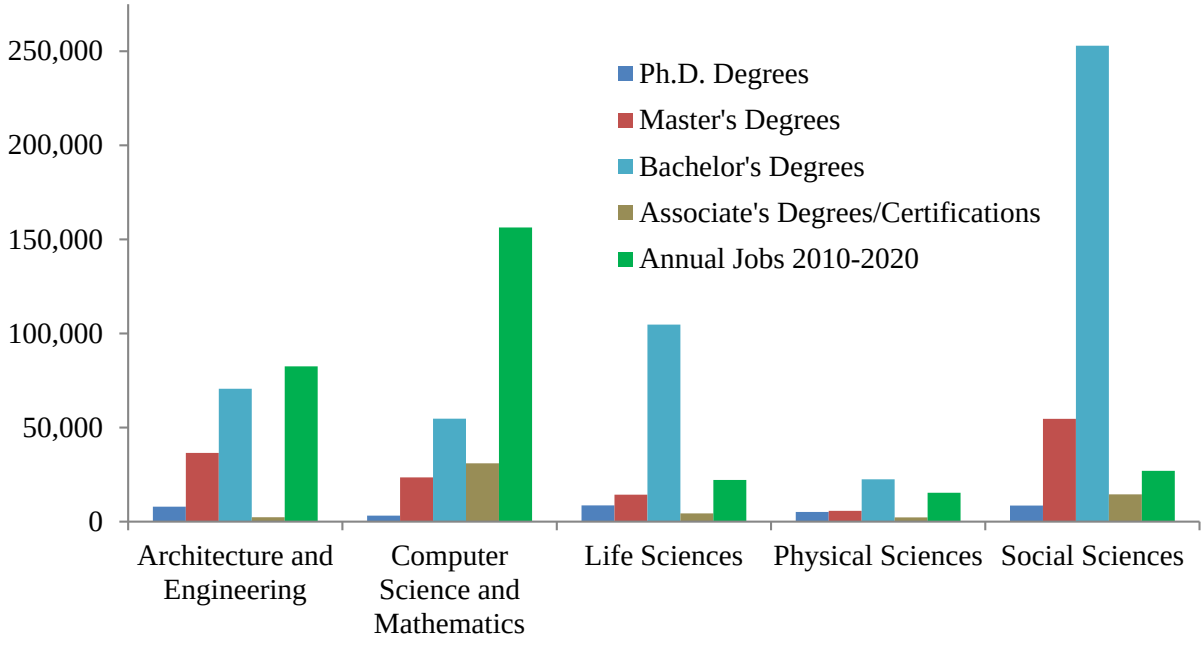

Figure 2.2: Annual degrees and job openings in broad S&E fields (2010-2020). Data taken from the Computing Research Association (cra.org). Used with permission.

The association for computing machinery (ACM) projects about 150,000 computing related jobs and only about 50,000 computer science graduates to fill them by the year 2020 [4, 5]. ACM projects computing to account for more than 60% of all science, technology, engineering and mathematics jobs (see Figure 2.3).

The Economic Policy Institute published that throughout the great recession (from 2008 to 2011), computer and IT unemployment remained well below the national average (around 6%) while average salaries remained fairly stable at about $75,000/year [6]. While some evidence shows this trend slowing [6], more evidence shows widely available jobs [4, 7]. Remaining competitive in the world technology market requires that the United States continue educating more computer and IT professionals.

In spite of the abundance of jobs, still fewer students than are needed choose to major in computer science. The association for computing machinery notes a deficient number of computer science students compared with the number of degree offerings (see Figure 2.4).

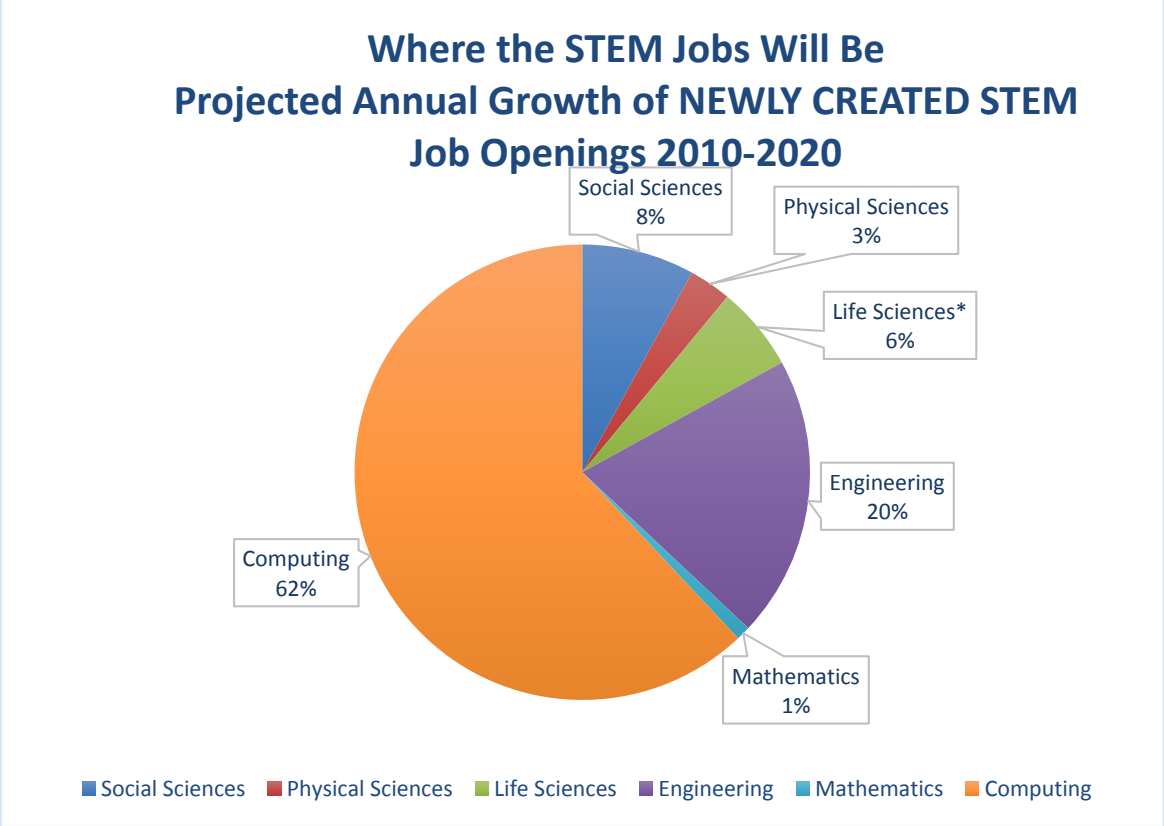

Figure 2.3: Where the STEM jobs will be. Data taken from the Computing Research Association (cra.org). Used with permission *Figure does not include medical related fields.

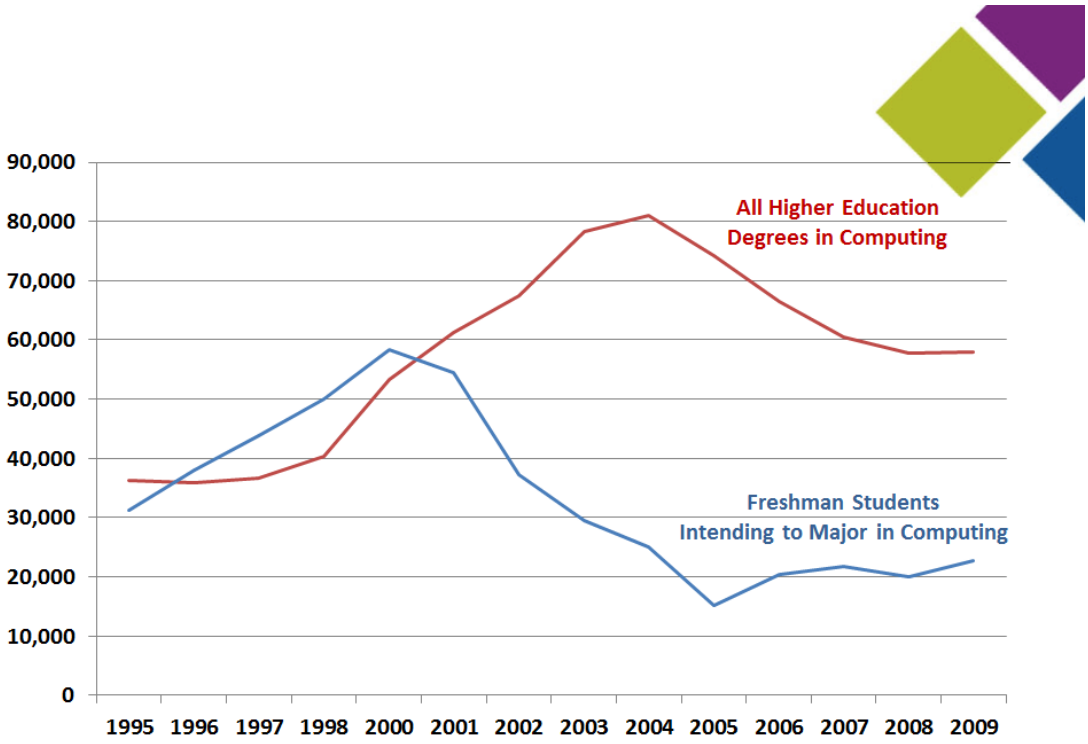

Figure 2.4: Higher education pipeline in computing. Slide originated from ACM; used with permission. *Data are not available from 1999.

We recommend a larger outreach to students thinking about a major. We recommend the following as ways to increase recruitment among students:

- Reaching out to potential students earlier than entry into college
- Combatting social stigmas that deter students from considering computer science
- Creating summer camps and other summer programs to pique interest among potential students
- Touting the successes and potential career opportunities for computer science graduates
- Reaching out to women

- Encouraging current graduates to contact potential students and talk about the benefits of a computer science degree. Contact by “near peers” has been found to be successful.

We will discuss each of these recommendations in detail in later sections.

CHAPTER 3

METHODOLOGY

This project began with several questions about computer science enrollment, namely:

1. Why is the enrollment in computer science programs so low compared with other STEM disciplines?
2. Seeing a greater need of computer science graduates, how do we boost enrollment rates?
3. Why is the enrollment of female students so low in computer science as compared to other sciences?
4. What are some ways we can attract more female students into the program and subsequently retain them?
5. Why is the attrition rate so high among computer science students that do enroll?
6. When are students leaving computer science?
7. What are some ways we can assist students in order to mitigate attrition?
8. What are the learning style traits of those who are successful at computer science? Are these styles different than students who drop out?

In order to answer these questions, we set up a series of surveys which we hypothesized would answer these questions and open up discussion at Utah State University and perhaps other universities as to how to best improve computer science education. We found that students generally leave in the first semester of the program [8, 9, 10]. We set up a set of surveys for first semester students to be taken at different intervals throughout the first semester (see Appendix A, B, C, and D). By doing this, we hypothesized we could pinpoint factors involved in attrition. The surveys were designed to be taken at intervals within the first month, second month, third month and fourth month of the semester.

We also surveyed two other groups: college seniors and professionals (see Appendix E, F, and G). The reason we chose to interview seniors is because we sought input as to why students stay in computer science. Professionals are surveyed because they give perspective on whether the programs are preparing students for the professional world, which is an important end goal of our study.

We gave the seniors and professionals a survey to answer the research questions listed above. We also used an additional survey to determine their learning styles based on studies by Felder and Spurlin [11, 12]. Our objective was to compare the learning styles of the three groups to see whether the learning styles of those who are successful are found to be much different from those who are just starting (see Appendix J).

Each first semester student was paid $25 for their participation. The seniors were paid $10, and the professionals were paid nothing. The money to pay them came from a grant by the National Science Foundation. A Utah State University Institutional Review Board reviewed and approved all surveys.

During the study, we found that we needed information from two other sources: high school women in computing and USU teaching assistants (see Appendix H and I). The teaching assistants help us understand the study habits of the first semester students and to verify some of their comments to us. The women in computing results were used to help with understanding some of the ways that high school aged women could be brought into computer science.

Several questions from each survey have less than the total number of respondents as participants were not forced to answer every question.

3.1 First Semester Surveys in Depth

The first semester surveys required the greatest degree of effort, simply because of their length. In First Semester Survey 1, we had 81 respondents. This number dropped off to 61 by First Semester Survey 2. We made attempts to contact these individuals to either finish this and the next two surveys, or to complete our First Semester Drop survey,

but none of these 20 responded to finishing anything additional. First Semester Survey 3 had 58 respondents. Again we made attempts to contact the three that decided not to continue and again received no response. First Semester Survey 4 had 56 respondents and again we attempted contacting the two that did not go on and received no response. Consequently, our First Semester Drop Survey did not receive any respondents.

We initially contacted many universities asking for assistance and we received productive responses from four, which are Utah State University, Brigham Young University-Idaho, Colorado State University, and Southern Utah University. Three of the four universities' professors offered extra credit for completing the surveys, which clearly made a difference in the respondent rate (see Table 3.1).

Table 3.1: Number of students per university finishing First Semester surveys

	1st Survey	2nd Survey	3rd Survey	4th Survey
Colorado State	5	4	3	3
Utah State	27	20	19	19
BYU-Idaho	33	23	22	20
Southern Utah	16	14	14	14

From implementing these surveys, we learned the following:

1. Incentives for classroom credit is worth more than payment in terms of getting students to respond.
2. Speaking to the individual professors was more effective than an administrator. The response rates were far less than we expected from Colorado State University. The university administrator we spoke to sent the email to hundreds of first semester students, but because the professors did not offer incentives, this was less effective.
3. We created a drop survey because we were very interested in those who

dropped the course, but we received no responses from these individuals.

We included several different universities in order to increase the number of possible respondents.

3.2 Senior Surveys in Depth

The seniors took one of two different surveys. All participants received $10. Utah State University students were also given class credit.

Once again class credit was a huge driving force. Of the Utah State University students that were given the survey, almost all of them completed the survey (see Table 3.2).

Table 3.2: Number of completions by college for Senior Survey

	Number of Completions
Utah State University	55
Colorado State University	17
Brigham Young University - Idaho	20
Southern Utah University	2

The professor we spoke with at Southern Utah University sent the survey on to another colleague who subsequently sent it to his students. Giving the survey to another professor with whom we had built no relationship resulted in less students at Southern Utah University than at the others (see Table 3.2).

3.3 Professional Survey in Depth

The professional survey had 120 participants completing the survey. This survey was given to the largest group of people to complete (1031 computer science alumni of USU were contacted) and consequently had the largest number of responses.

3.4 Tutor and Women in Computing Surveys in Depth

The tutor survey was conducted in order to get more feedback about student

attitudes at different times during the semester and what the tutors see as needed improvements. 10 tutors answered the survey and were paid for their efforts.

The women in computing survey was completed by high school age girls in the state of Utah who had received recognition from the National Center for Women in Technology Aspirations Award. There were 28 responses from NCWIT top candidates.

3.5 Learning Styles Surveys in Depth

We also utilized a learning styles survey. We had the following response rates for the learning styles surveys:

- For professionals we had 42 respondents
- For Seniors we had 55 respondents
- For first semester students we had 81 respondents

CHAPTER 4

RESULTS

We break up the discussion of results into two major sections related to increasing computer science graduation rates: Retention and Recruiting. Both are necessary and valuable to improve quality programs.

4.1 Recruiting More Students

In the following sections, we discuss why students are not joining computer science as well as some recommendations to assist students to come into computer science.

4.1.1 Reaching Out to Students Earlier

Knowing when students choose their major allows us to focus our recruiting efforts on this time period. In our study, we found that many students perceived a need to be introduced to computing earlier. As one student we surveyed said, “Get elementary students to enjoy science and math. As it is, most children are completely uninterested in these topics. There's no quick and easy way to boost the number of engineers in the nation.” Another student agreed, “Introduce people to computers and programming earlier.” Other students and professionals added a timeframe to introducing people to programming by saying they need, “More exposure to it during high school.”

It is possible that by the time high school arrives, many have already selected a major. We know that we need to start by at least high school to help get more students involved in computer science. Starting earlier would also be helpful in persuading students to choose computer science as a major.

Lack of funding for and the importance placed on computer science is causing a decline in the portion of schools that offer introductory computer science courses; consequently, fewer students are taking high school courses in computer science [13].

Companies like Microsoft and Google are reaching out to high school students by both creating conferences and supplying instructors to high schools in various locations throughout the country [14]. This outreach is in response to the dearth of available people to fill the jobs that these companies have. Universities can also provide such outreach programs at an earlier age.

Although an increasing number of economic specialists express concern about a direct link between technology, innovation, and national economic survival, few educational policy leaders understand the profound need for computer science education at the high school level. Despite students' need to incorporate computer science skills in almost every industry, most high schools in the United States fail to recognize computer science as a core subject such as mathematics [15].

The decreases in computer science courses offered in high school result in less exposure to computer science, which leads to less interest on the part of a capable student. Many universities have addressed this problem by offering a short course or camp during the summer [15, 16]. These camps introduce students to the fundamentals of computer science while making it a fun experience. Summer camps serve to pique the interest of a few high school students who might then learn more on their own or talk with their friends about the subject.

We asked senior students how to increase those participating in the major. Twenty percent responded that we need to advertise more or have more outreach programs (see Table 4.1.1). Many mentioned the need to go to high schools and advertising there. One student gave this response, "They need to be exposed to it. So, how do we do it? If we're serious about it, we go do assemblies at high schools or at college fairs or whatever and we introduce them to it."

Table 4.1.1: (Open-ended) Ways to increase number of students (N=69)

In your opinion, what would be the best way to increase the number of students in your field of expertise?	Percentage
Provide more exposure at High School or earlier	20%
Focus on what is relevant in the industry	20%
Give students something more interesting to do	16%
Clarify career opportunities	12%
Overcome Stereotype	6%
Give comparative pay statistics	4%
More CS course options	4%
Cap H1b visas	4%
Better teachers	4%

From Table 4.1.2, an interesting perception is that many students do not know what computer science is like and because of this do not choose the major. A recent study by Point Loma Nazarene University indicated that 80% of non-computer science students did not have any idea what computer science majors learned. Of those that indicated they do know what computer science students learn, most believed the major focused on computer programming. One result of the study is that only 2% of high school students surveyed had any reasonable grasp of what computer science entailed [17]. A large percentage of students in this study (50%) were opposed to computer science because they did not want to sit in front of a computer all day, as if that is all you can do with a computer science degree.

Table 4.1.2: (Open-ended) Why it is thought others do not join the major (N=84)

In your opinion, why don't more students pick the major you chose in college?	Percentage
It's hard	30.4%
Stigma	18.8%
Don't know what it's like	14.5%
Math	8.7%
Intimidating	8.7%

Lack of knowledge about computer science is perpetuated by the lack of computer

science programs and/or requirements in high school. The study by Point Loma Nazarene University spoke about the fact that few high schools require any computer science; some do not even have a computer science program. This has been verified by many articles and papers indicating that the computer science programs are decreasing in number in the United States because of funding and a misunderstanding of what computer science education is [13, 18, 19]. There is little recognition of computer science as a scientific discipline distinct from mathematics or technology training [20, 21].

In 2004, the Computer Science Teachers Association (CSTA) surveyed 14,000 high school teachers who defined themselves as computer science, computer programming or AP computer science teachers. In spite of computing's growing importance in society today, only 26% of responding schools require students to take a computer science course and only 40% offer Advanced Placement (AP) classes in computer science. Among students who took the introductory course, only 32% were female and in AP computer science courses, only 23% were female (actually decent numbers when compared with college enrollment [22]). These data are supported by other, more recent data from the College Board that finds while AP testing in other disciplines increased by 19% overall, participation decreased by 8% in the computer science A exam and 19% in the computer science AB exam. Currently about 2% of college-bound high school students take the computer science AP exam (either the A exam or AB exam) [19].

The high school computer science teachers indicated that the greatest impediment for students enrolling in computer science was not the difficulty of the subject or even that the subject was considered "geeky," but the amount of time in the student's schedules. Anecdotally, one study reports that students are also increasingly being dissuaded from computer science because the belief is that computer science is no longer a source of rewarding and varied career opportunities, based on media reports rather than marketplace realities [23].

We recommend creating computer science summer programs for high school-age

students. It is important to consult with high schools on how to improve their computer science programs and attract more students to these programs such as the programs by Microsoft and Google.

4.1.2 Overcoming Social Stigmas

Why aren't more students attracted to computer science? Labeling computer science as geeky or nerdy could be to blame. Despite rampant distribution of gadgets, smartphones, and tablets, computer science students are still largely seen as geeky [24, 25]. Our study reveals that this stigma may be a deterrent.

We asked seniors an open-ended question as to why they think more students do not choose computer science as a major. The second highest response they indicated was that there is a stigma attached with those in the major (see Table 4.1.2).

Every group we surveyed indicated that the stereotype of being a nerd or geek is still an issue when deciding whether to choose computer science as a major. One student said, "They are scared away by all the nerds." Another indicated that, "It has a negative stigma of being just for socially awkward smart kids that wear pocket protectors." Yet another mentioned, "Since computer science is associated with nerds, [I'm] assuming that they don't want to be labeled/viewed as such."

The stigma attached to computer science is well documented [26, 27, 28]. What is less clear is what to do about the stigma. A study done at Waterloo focused on a seminar targeting 14-16 year old girls [27]. At the beginning of the seminar, the girls were asked whether they plan to take computer science and 30% responded affirmatively. At the end of the seminar this had jumped to 55%. It follows that a greater degree of information, presented in a creative way, does change minds and debunk stereotypes. We recommend similar seminars for high-school age or younger students.

Another way to overcome social stigmas is to have those already working in the field speak out about what computer science really is. Posting videos of current computer science graduates talking about their careers could also result in boosting recruitment. Most professionals we surveyed are either extremely or very likely to

recommend their field to others (80% combined; see Figure 4.1.1). At the same time, most of them almost never or only occasionally talk about their field of work (67% combined; see Figure 4.1.2). A video promoting learning how to program using professionals currently in the field is available at code.org. This video features many famous programmers promoting programming and learning computer science. Such a video is exactly what our research indicates should be utilized. We recommend using this video at the engineering and computer science websites.

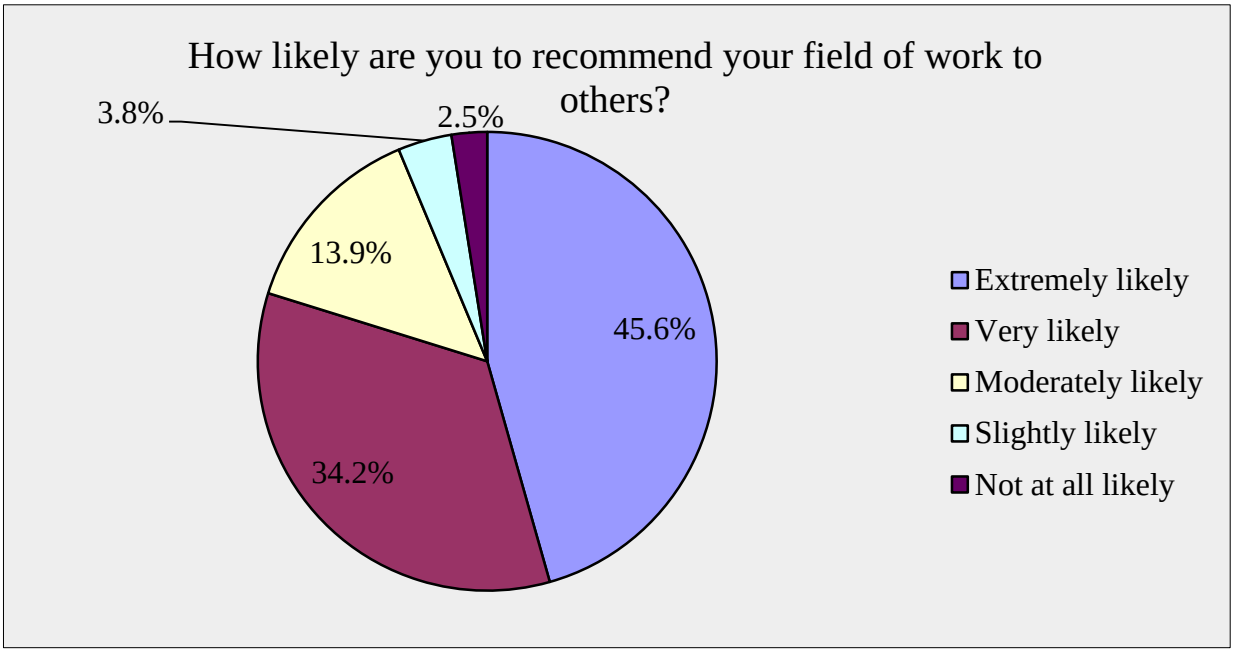

Figure 4.1.1: How likely professionals are to recommend their field of work to others (N=79).

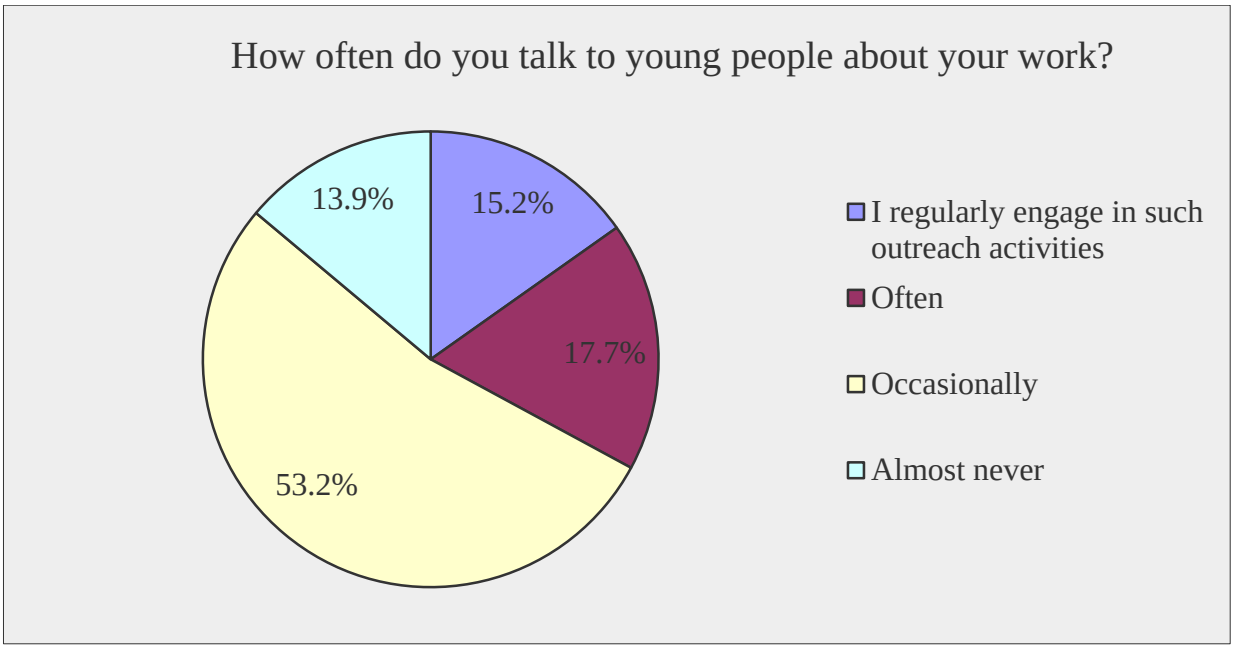

Figure 4.1.2: How often professionals speak about their work to young people (N=79).

4.1.3 Considering Necessary Math

In our surveys, many indicated that the math pre-requisites are also a great deterrent for students joining computer science. Professionals in our survey observed that most computer science related fields do not need the level of math that is required to graduate from most computer science programs. One professional mentioned that students and teachers, "need to understand that most jobs, you'll just be doing algebra, and simple math equations." Is it necessary for each student to have passed calculus II in order to be a successful programmer? Some students who do not have the math pre-requisites necessary to take computer science could be discouraged by the long sequence of prerequisites.

We asked computer science tutors in our program their thoughts on mathematics in the computer science courses. One tutor said, "I see uses for discrete math all the time but I have yet to see hardly any calculus [used]." Another said, "To me it appears that the upper level levels [of math] are not really applicable." Are we deterring students from the major who would, without the stringent math requirements, be successful? We need to look at the level of math that is truly necessary for each computer science course and make those courses

the requirements for the overall computer science major.

We discuss in more detail the possibilities for adjusting computer science mathematics prerequisites, including ABET accreditation standards, during the retention section.

4.1.4 Creating Out of School Camps and Programs

Summer programs are one way to generate interest and increase awareness among high school age students. These programs are being used to prepare students for computer science in college as well. Dr. Luczaj conducted several of these summer programs in Ohio during the summers of 2007 and 2008 and reports wide success. During these two summers, he reports that 90% of students agreed that the overall impression of the conference was positive [15].

He goes on to report the methods to create a successful computer science summer program. He emphasized hands-on learning. The first project was an animated logo design created by the student. It was common for the students to work through their morning and afternoon breaks because they were so absorbed in their projects. Those who produced the projects mentioned creating something that should be an academic challenge for the students. They structured it around the philosophy that the students should learn and experience things unavailable to them in during the school year. The successful program also was created based on these two ideas:

1. Material would be broken into small units with tangible results
2. Material would be accessible to the students, with examples related to the students.

As a consequence to these principles being followed, students reported things like, “I was treated like an adult” and “the course content was much harder than high school, but [it] was more interesting and fun.”

Rivier College in New Hampshire wrote about their successes and also failures in offering a summer program. One interesting thing brought up by this program was the fact that there are a lot of prestigious universities that offer these computer science summer camps which they offer as a recruiting tool for highly sought after students [16]. These summer programs could also be used as recruiting tools for many universities.

Private companies are already using summer camps and other outreach programs as both educational tools and recruiting tools. Microsoft and Google both have outreach programs. Google currently has an annual Computer Science Summer Institute. Microsoft encourages its employees to volunteer at high schools who need extra computer science expertise [14].

We recommend creating the programs that also appeal to non-computer science types in order to attract them into the program as well. These summer camps could also be used to dispel the stereotypes that are common among those in computer science majors [27].

Winners of the Utah NCWIT competition (a program honoring high school girls) indicate that their interest in computing began early (some as young as age seven). While they did much exploration independently, usually it was a relative or a summer program that served as the gateway to computer science.

Table 4.1.3: Pre-college women computing experience (N=28)

Which of the following computing activities/concepts have you learned, or participated in, either through school, in an after-school/summer program, or on your own? Tell us how involved you have been by using the rating scale: Not at all, A little, Pretty much, A lot.

Answer Options	Not at all	Only a little	Pretty much	A lot
Word Processing (e.g., MS Word, WordPerfect, Open Office)	0.0%	3.6%	3.6%	92.9%
Spreadsheets (e.g., MS Excel, Open Office, IWork)	3.6%	17.9%	21.4%	57.1%
Presentations (e.g., MS Powerpoint, Keynote, Prezi, Open Office)	0.0%	0.0%	17.9%	82.1%
Desktop Publishing (e.g., MS Publisher, InDesign, Pagemaker)	28.6%	32.1%	14.3%	25.0%
Mathematical Applications (e.g., MatLab, Mathcad, Mathematica)	44.4%	33.3%	14.8%	7.4%
Web Design/Development (e.g., Dreamweaver, MS Expression, Web Studio)	25.9%	22.2%	22.2%	29.6%
Multimedia Design (e.g., Photoshop, Imovie, Premiere, Movie Maker, Illustrator)	3.6%	17.9%	25.0%	53.6%
Computer Aided Design (e.g., Autocad, Inventor 3D)	48.1%	25.9%	14.8%	11.1%
Game design (GameMaker, Blender, Maya, Visual Studio/Basic/C++, C#)	46.4%	25.0%	14.3%	14.3%
Mobile application development (Objective C, AppInventor, Iphone/Android, Ipad/Tablet)	35.7%	35.7%	10.7%	17.9%
Web application development (Facebook Api, Flash, Java, HTML5, PHP)	28.6%	21.4%	21.4%	28.6%
Computer Technology (e.g., tech crew, install/maintain computers for yourself and others, software installation)	10.7%	21.4%	25.0%	42.9%
Network/System Operations, Maintenance (e.g., system manager for multiple machines)	51.9%	29.6%	11.1%	7.4%
Network design/development (e.g., LAN, wireless)	48.1%	29.6%	14.8%	7.4%
Network security	51.9%	33.3%	11.1%	3.7%
Robotics (e.g., Labview, ROBOTC, LEGO Mindstorms)	71.4%	14.3%	7.1%	7.1%
Graphic (e.g., Scratch, Alice, Flash, OpenGL, Java 3d)	28.6%	32.1%	25.0%	14.3%
In a coding language (e.g., Visual Basic, Python, Java, C++)	14.3%	21.4%	21.4%	42.9%
Using variables, loops, decision logic, arrays/lists	25.0%	7.1%	32.1%	35.7%
Using advanced data structures (stacks, queues, trees)	64.3%	32.1%	3.6%	0.0%
Mainframe programming	67.9%	21.4%	3.6%	7.1%
Database design/programming (Access, MySQL, Filemaker)	53.6%	35.7%	10.7%	0.0%
Open source contributions (contributing original code segments to open source libraries)	75.0%	10.7%	14.3%	0.0%

Summer programs do not need to introduce new software or languages to be effective. As we see from our NCWIT poll (see Table 4.1.3), most of these highly motivated students have had experience with a spreadsheet software program (e.g. MS Excel). Without having to teach an entirely new software program to students, this familiar software could be used to teach computing terms and concepts such as Booleans, macros, looping, formatting and sorting.

A similar approach to creating summer programs would be to create technical clubs in high schools. This could be a low investment way to give students an extra-curricular experience in computing. In the NCWIT poll cited earlier, some students had experienced extra-curricular clubs, but many others had not. Many schools do not have computing clubs, which could be a way to increase interest and mentor students in high school.

Formal coursework and organized summer programs are definitely important. However, with limited expertise in computing and many districts offering few advanced courses in computing, it is important to also encourage independent exploration. Applicants to NCWIT Aspirations Awards indicated a host of activities they had pursued independently. The following indicate the importance of providing extracurricular opportunities in computer science:

- My all-girl team won the iGEN LED Challenge in which the teams designed, wired, and programmed colorful LED displays to highlight something important in their community. We created a holiday tree for the Festival of Trees, an auction to benefit Primary Children's Medical Center.
- I edit the pictures I take with photo editing programs. I have also created videos from my photos, presenting them at group gatherings and parties.
- In Jython, I learned how to manipulate graphics.
- With Excel, I have been making time cards, and tracking employee information for a local company.
- I set up the high school library webpage and Facebook.
- I won the schools advertising and design competition in Adobe Illustrator.

- I used the Premiere program to make a short music video with animated words, pictures, and music.
- I competed in a national cyber forensics competition, involving encryption, operating systems, and file systems.
- CAD software was used to design a floor plan.
- I discovered MS Publisher in 5th grade, making silly newsletters and birthday cards
- I created an iMovie project, consisting of public domain footage, interviews, music, narration, voice over, text, slow motion effects, and sound effects. Currently I've been learning Photoshop and Dreamweaver. App inventor, C++ language.
- Using AppInventor, I program Android apps.
- I started learning about JavaScript with Codecademy

In many cases, a relative or friend provided an "invitation" for the exploration of computing principles. Because such invitations are random, many youth are missed. We encourage the development of a repository of interesting projects that youth have created. Such a repository could serve as motivation for others as well as recognition for the creators.

4.1.5 Using Successes in Computer Science as a Recruiting Tool

One way to successfully recruit students into a major is to tout what the programs are doing well. Reviewing our survey, we found some good things that are worth highlighting.

Students generally indicate that the computer science programs we surveyed are doing well. 81% of students indicated they are willing to help other students when they need help (see Figure 4.1.3). This helps us get a general idea of the students' attitudes toward their peers. It also helps to know that students are willing to help each other if a collaborative project is assigned.

We asked the first semester students whether other students in their class treat others with respect, are interested in others, or treat others with contempt and 74% of students said they feel like they are treated with respect. In some programs,

antagonistic behavior has been reported, but in the programs we surveyed the majority of students are interested and respectful.

Students in the first semester generally feel like their skills in computer science are strong; as indicated by 67% of the students. Eighty-eight percent indicated that the homework is interesting. It follows that if the homework is interesting, the student will spend time on the homework, thus making their skills stronger in the assigned material.

One thing that we find odd is that 62% of students indicated they are getting either A's or A-'s. Though the first semester course is intentionally less intense than those courses that follow, this is still a fairly large number of A's and A-'s. Sixty percent of students also indicated that their education is more effective than at other institutions. It is interesting that most students think highly of the educational institution in which they are enrolled.

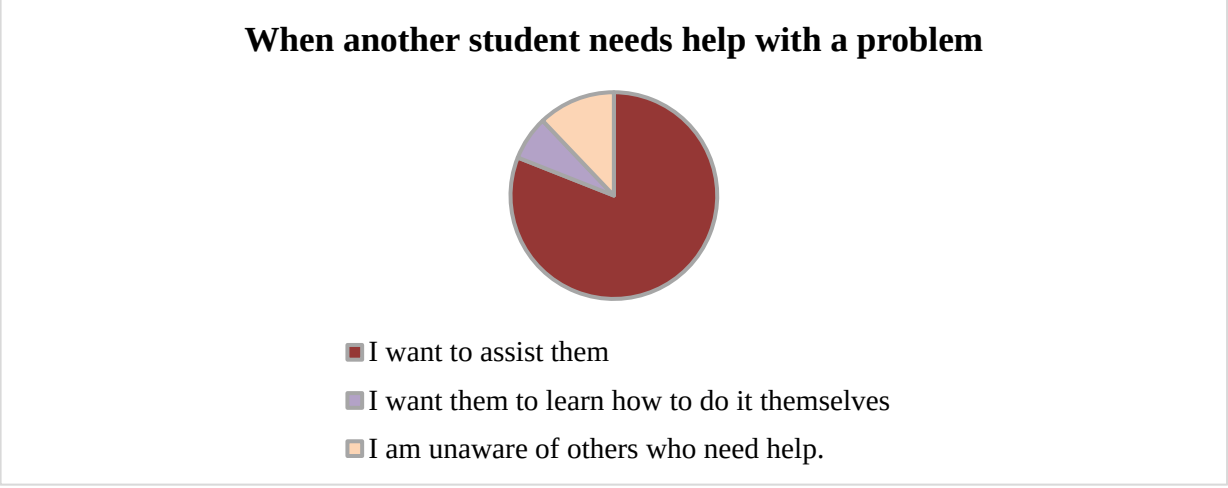

Figure 4.1.3: First semester students desire to assist other students (N=58).

Sixty percent of first semester students indicated that they only spend about 3-6 hours per week outside of class on the homework and study. This means that the workload is probably about right in the first semester. We could consider use of this as a metric to make sure that we are not overworking our students in subsequent semesters.

Ninety-seven percent of students said that the first course is useful. Sixty-three percent of students also indicated that they would recommend the class to others. We recommend using these statistics as additions to current recruiting information.

4.1.6 Reaching Out to Women

Computer science programs throughout the country struggle to get women into the major. Computer science remains well below the other sciences in terms of the ratio of women to men [3]. Attracting women to computer science could be a great way to increase the number of majors. The National Center for Women & Information Technology conducted a study on the reasons for choosing a computer science major for men and women. The results (see Figure 4.1.4) show very little interest in computer games among women as compared to the men.

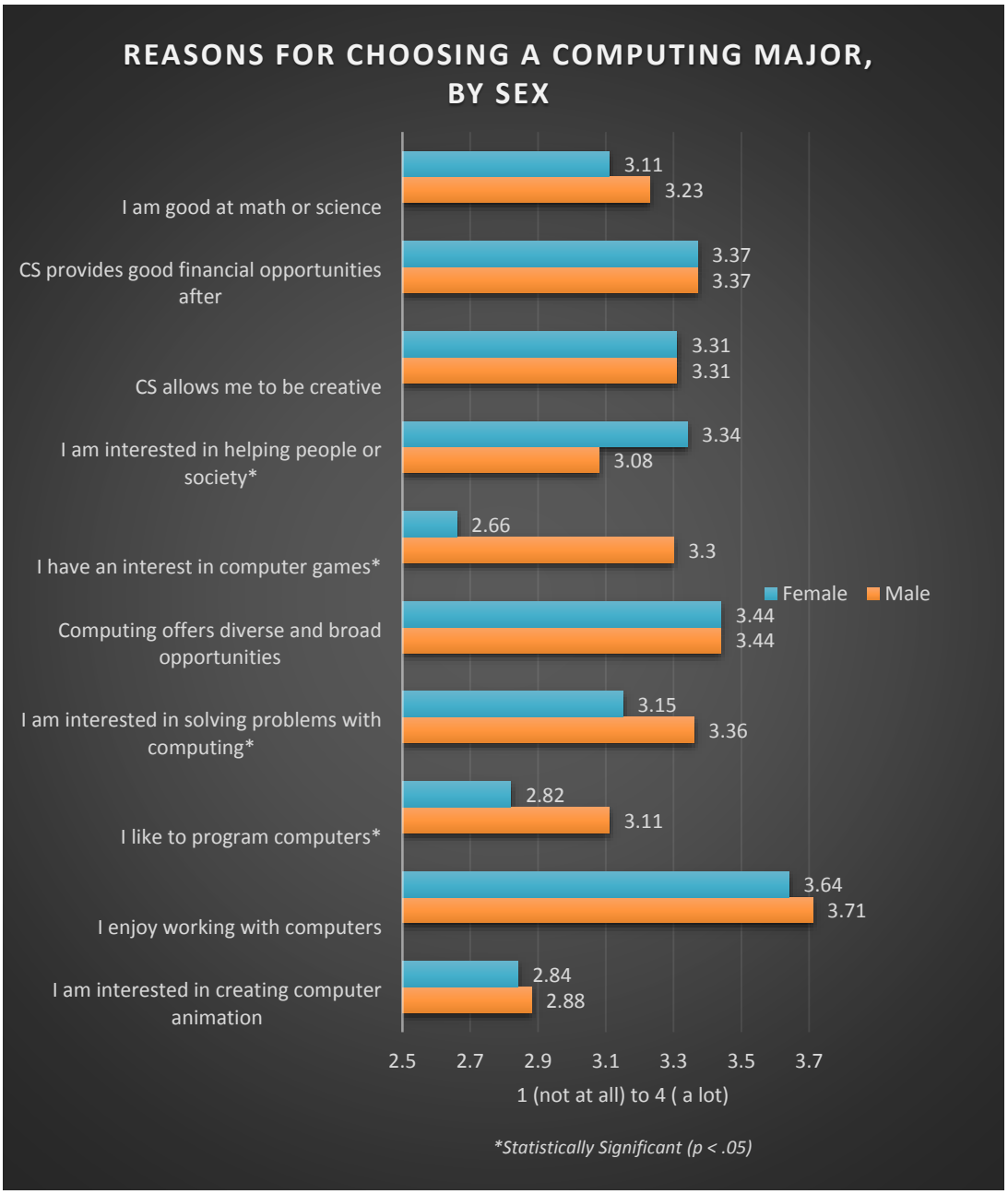

Figure 4.1.4: Reasons for Choosing a Computing Major by Sex. NCWIT. Used with Permission (N=1,434 introductory computer science students).

Women show more interest in helping people or society than their male counterparts (see Figure 4.1.4). Traditional computing assignments could be a great deterrent for women, which make up roughly half the general population. In order attract

more women into computer science, we need to highlight computer science as being meaningful in society.

In polling top Utah applicants for the National Center for Women in Computing Aspiration Award (for High School aged women), 82% indicated they had done work with presentation-type software (e.g. MS PowerPoint, Keynote, etc.). In the same study, 54% of women indicated they had worked with multimedia design software (e.g. Photoshop, iMovie, Movie Maker, etc.). All women surveyed are interested in completing a computer related degree in college. One way we could increase computer "likability" with female students would be to increase work with presentation type or multimedia software. Very few women indicated they had done any work with game design. It follows that the game assignments we think of as fun and interesting for students, might have limited appeal to women.

Many tend to think that programming is the entry point for computer science students, but a surprising number of women in this study mentioned they are interested in computing but had done little programming. Creating attractive courses designed to pull people into computing that are not programming intensive may be one way to attract both women and men to the major.

Women currently make up less than 15% of all computer science students [29]. Making computer science more female-friendly should be a top priority at computer science programs around the country (because females make up over half the college population). According to a recent report put out by the Association for Computing Machinery, reaching out early to girls is effective in overcoming the "geekiness" stigma attached to computer science. An all-girls middle school in California requires computer science for 3 years. In this school, students are taught a variety of real-world applications such as designing a traffic intersection near the school. Because the school introduces computer science before the stigma is attached, the girls are attracted to computer science.

Ninety-four percent of students we surveyed agreed with the statement that,

“Women can excel in careers that involve computing.” This seems to be a common sentiment: women can do it. This sentiment does not translate into large percentages of women going into the career, however.

Our NCWIT study reveals that the top three female computing career plans include computer graphics, programming, and general computer science. In another section, 78.6% mention experience with Photoshop and other multimedia design software. The spread of females with web design experience was about even across four categories ranging from “a lot” of experience to “No experience at all.” We could attract females to the major using graphical software they are familiar with and then teach them programming using web development, which heavily uses graphical software for webpage mark-up.

Seventy-eight and one-half percent of female students have experience with spreadsheet software (ex. MS Excel). We could use familiar tools such as this to teach computer science principles like loops and conditional statements without needing to introduce them to a completely foreign integrated development environment (or IDE).

Looking at what females explore in their spare time gives us a clue as to how to attract more women into the major. Avoiding those activities that female students do not participate in and increasing the emphasis on those they do participate in could prove successful. For example, Figure 4.1.5 from our study shows that a lot of high-school age women chose to participate in college level courses. Thus, we could increase the number of college level courses available to them. Another cost effective way to increase visibility would be to have a technology related high-school club or team.

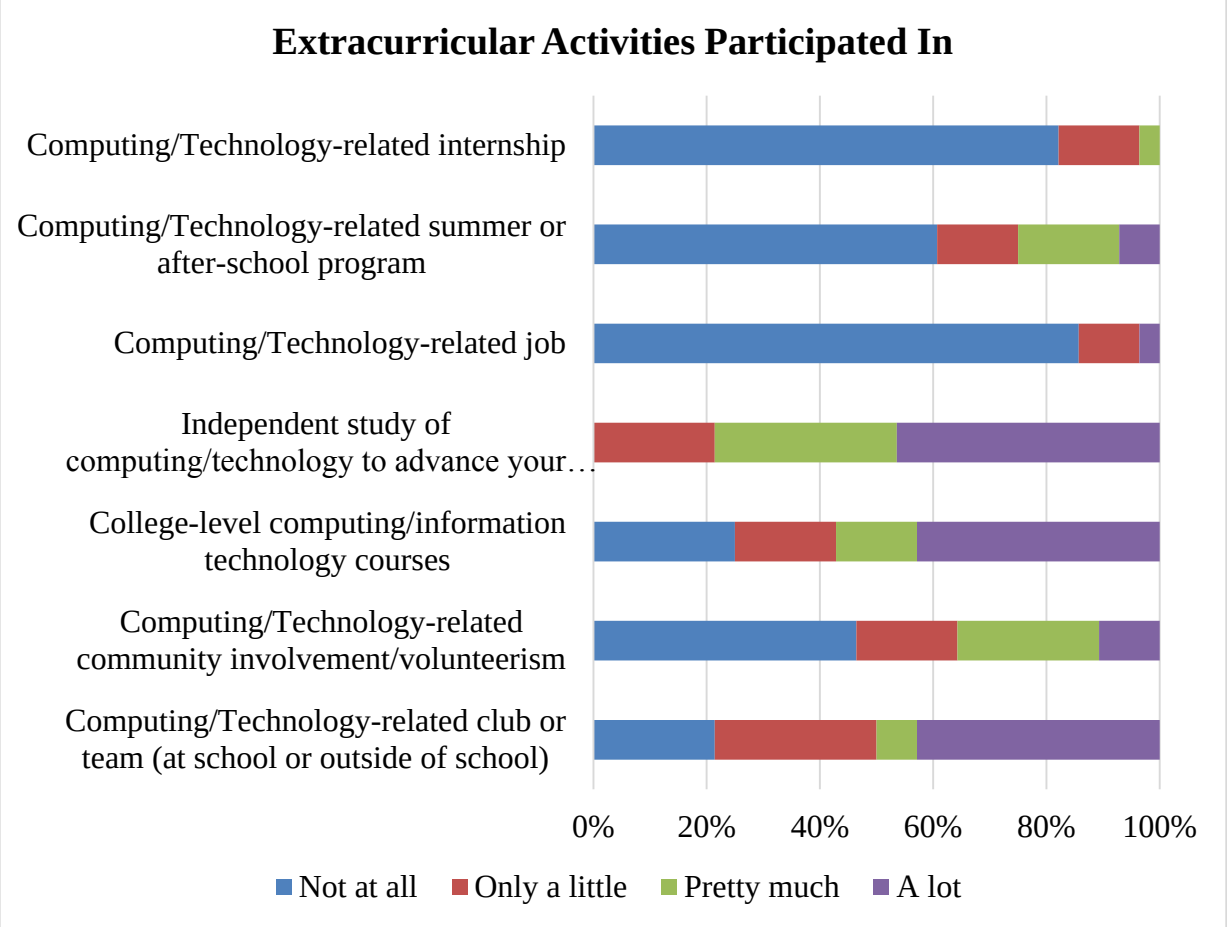

Figure 4.1.5: Extracurricular Activities of women in computing from our NCWIT survey (N=28).

4.2 Retaining Current Students

We need to keep students (who take a computer science class) in the major. A typical attrition rate in computer science classes is between 40% and 60% [8], however, sometimes the attrition rate is even higher. At Southeastern Louisiana University the attrition rate was more than 90% [28]. One study found that the attrition rate during the introductory course at the University of Illinois at Chicago can be as high as 50%. The authors note that computer science departments today are not successfully attracting a diverse range of students to computer science courses and they are not engaging those students who do enroll [8]. These authors also state that addressing the factors

contributing to women leaving the introductory course may also increase the number of men – in particular, those men who are not currently retained in the major.

Given that attrition is a problem in computer science throughout the United States, what can be done to address this problem? We discuss several areas where colleges could improve that we found from both our surveys and related works:

- Address teaching and learning style differences between teachers and students
- Create a pre-introductory course for students with little to no background in computer science
- Adjust math requirements to better fit what is actually needed in computer science
- Create more meaningful assignments to engage students
- Create a non-weed-out learning environment
- Improve instruction quality by changing evaluation techniques

As we address these issues, it is our hypothesis that retention will increase as students become more comfortable in computer science.

4.2.1 Teaching Styles

One of the most important characteristics of an instructor is being able to effectively teach the required material to the students. In order to learn whether that teaching is being effectively received, we surveyed first semester students from four universities, two of which are research institutions and two are teaching institutions.

We asked several questions regarding the teaching styles of respondents' professors using an open ended format. Both seniors and professionals indicated that in some cases the professors' teaching style was undesirable. Professors' teaching styles was the fourth least desirable thing about the major (at 10% - not shown in a table) and fifth highest (at 7.4%) for seniors (see Table 4.2.1). When asked how to improve the

education provided (this question was also open-ended), 9.6% of the professionals mentioned that we should hire better teachers. One student said, “My CS2 professor almost ruined the Major for me in CS2…” This quote from the data brings up a question: What are some of the reasons students perceived teachers to be the problem? Understandably most universities have research as an emphasis for their faculty. As a result, teaching can suffer [30]. However, it is unclear whether it is a teacher problem or a student problem.

Table 4.2.1: (Open-ended) Least desirable things about major from a senior perspective (N=84)

What are the least desirable things about your major?	Count	Percentage
Takes a lot of time	31	25.6%
Math	15	12.4%
A lot of work	14	11.6%
Meaningless courses/assignments	11	9.1%
Professors' teaching styles	9	7.4%
Steep learning curve	8	6.6%
Male-Female ratio	6	5.0%
Dealing with Stigma	6	5.0%

This also raises the question of what being a good teacher means in computer science. A teacher who merely gives the code to their students is not teaching the students how to think, although this teacher may be getting great reviews from their students as a result. Tabulating the comments from students about the impact of teachers, we find that most instructors are doing a good job.

Some supplemental comments from the professionals are helpful in explaining in more detail what being a good teacher means and how best to teach computer science

students. At the end of the professional survey, we asked the question, “Do you have any other comments about the CS department?” to which one professional responded about the learning atmosphere unique to computer science, “I enjoyed the professors that coded with us (it could have even been on the white board) more than those with slides to go through. Slides kill learning!”

Table 4.2.2: (Open-ended) Improving education from a senior perspective (N=77)

What suggestions do you have for improving the education provided?	Percentage
More hands-on activities	19.4%
More real-world examples	16.1%
Restructuring entry-level CS courses/Giving more background to beginners	14.0%
Professors being open to alternative teaching methods/allow one-on-one time with students	11.8%
Hire better teachers	9.7%
No written tests/better tests	5.4%
Professors need to be more realistic about abilities	5.4%

In our study, we asked students how their courses are generally taught. Most (75%) indicated that there are more lectures and few, if any activities or demonstrations (see Figure 4.2.1). Coupled with the data from Table 4.2.2, we realize that students are not getting the hands-on experience that they feel would be most appropriate for their education.

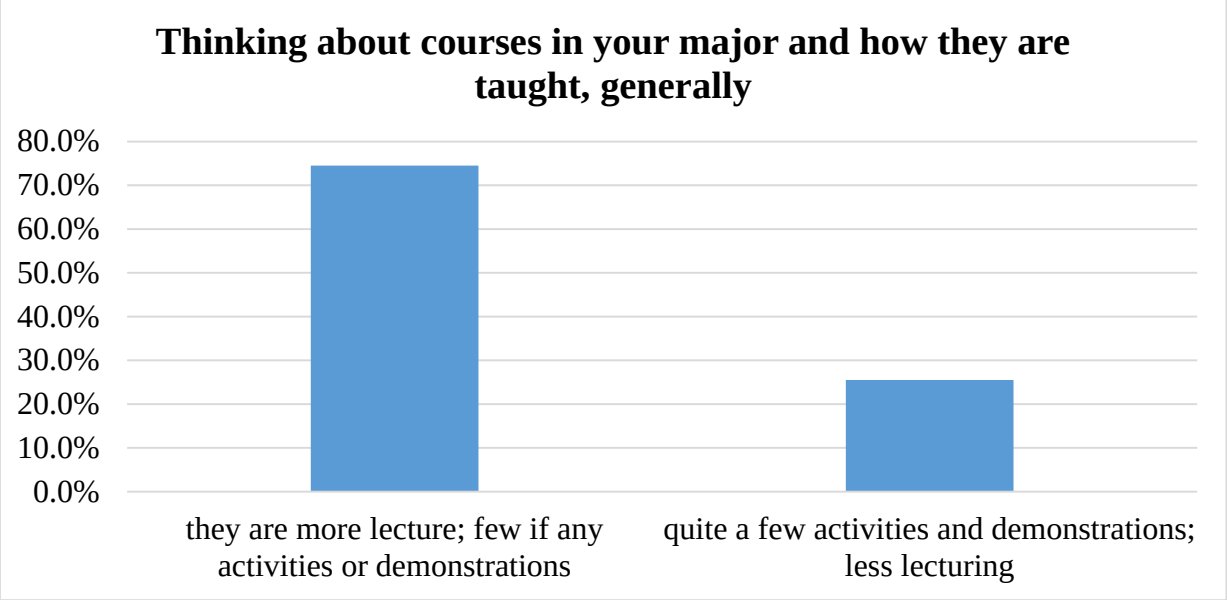

Figure 4.2.1: How students are generally taught (N=51)

There has been much research done on the correlation between a good teacher and the students' reception of the material [31]. We will not belabor the point here, except to say that if the instructor understands the needs of the students, then it could lead to better teaching. One student talked about the trouble with expecting most students in the major to be those who have been exposed to the concepts for a long time. This student said, "We want to learn to program well but it is harder to do so when every professor treats the subject like we have been coding since age 12." Another spoke similarly, "[teachers need to give] better explanation of what we are doing in our in-class demonstrations."

One recommendation that is being used at universities throughout the country is a Small Group Instructional Diagnosis developed at the University of Washington [1]. The key to this approach is to keep the students anonymous. Briefly, a Small Group Instructional Diagnosis (SGID) is a midterm diagnosis on how teaching is being received. The evaluation is conducted by a staff member who is not associated with the class. At the University of New Mexico, this has been a way of improving instruction that has been

successfully employed [32]. The University recommends that students answer between 3 and 5 questions individually. Then the students are separated into groups of 3 to 5 students, wherein the students discuss the answers to the questions. Once a consensus is reached in the group, this answer is recorded. The answers are thus made as a group and are not indicative of just one individual experience. Some of the recommended questions are:

Q1) What do you like MOST about this class and class instruction?

Q2) What do you like LEAST about this class and class instruction?

Q3) What suggestions do you have for your INSTRUCTOR to improve your learning?

One reason this would be more useful than a traditional teacher evaluation is because of the group nature of the questioning. Traditional teacher evaluations are more subject to change based on the whim of a single student or a set of circumstances outside the control of the instructor. One student's experience, good or bad, can have an impact on the overall quality of responses. Other reasons traditional evaluations are not sufficient include:

1. Individual students could have stress outside the class that affects the student's learning experience and prompts a low score
2. A student's and teacher's learning and teaching styles could be out of line and prompt a low score
3. Instructors could be too easy on the students and therefore be given an inflated score for teaching that was not necessarily instructive
4. Instructors personalities could become problematic and despite being a great teacher, could be given a low score

One difficulty for faculty is the conflicting suggestions given in teacher evaluations.

Group suggestions are therefore valuable as a common set of suggestions is derived.

In order to check the value of traditional teacher evaluations, we plotted the improvement over 6 years of teacher evaluations at Utah State University. Of the 21 instructors we evaluated, for 10 of them, the scale remained flat, for 5 of them the scale increased slightly and for 6 of them the scale decreased slightly (we show 2 examples in Figure 4.2.2). Considering most instructors try to get better at teaching, this informs us as to the limited value of these evaluations.

One deterrent from using SGIDs has been the amount of time away from class this takes the students and the evaluator, but as Hult puts it, “What are our reasons for evaluating, anyway? What do we hope to accomplish?” [32]. Before implementing the SGID, universities should ask: do the benefits outweigh the cost? Teachers who have participated in SGIDs certainly think so. In a study Bowden did on SGIDs, one instructor said, “It confirmed what I’d already thought was going on in the class—positive and negative—and gave me concrete ideas for ways the course could be improved, most of which I implemented. My students also seemed to appreciate the philosophy of the SGID. Not only did it give them an opportunity to air their concerns about the course, but it also provided a forum for discussing these issues” [32]. Other teachers reported improved classroom discussion, overall increase in participation, better attendance, boosted camaraderie, and mutual validation. One teacher summed it up, “It’s about time that idiotic space between students and teacher—in the context of pedagogy—is reduced.” It is indeed.

The SGID also benefits the students. Bowden’s study mentioned that it is good for the students to get a “gripe” session so that the students feel comfortable in discussions with the facilitator.

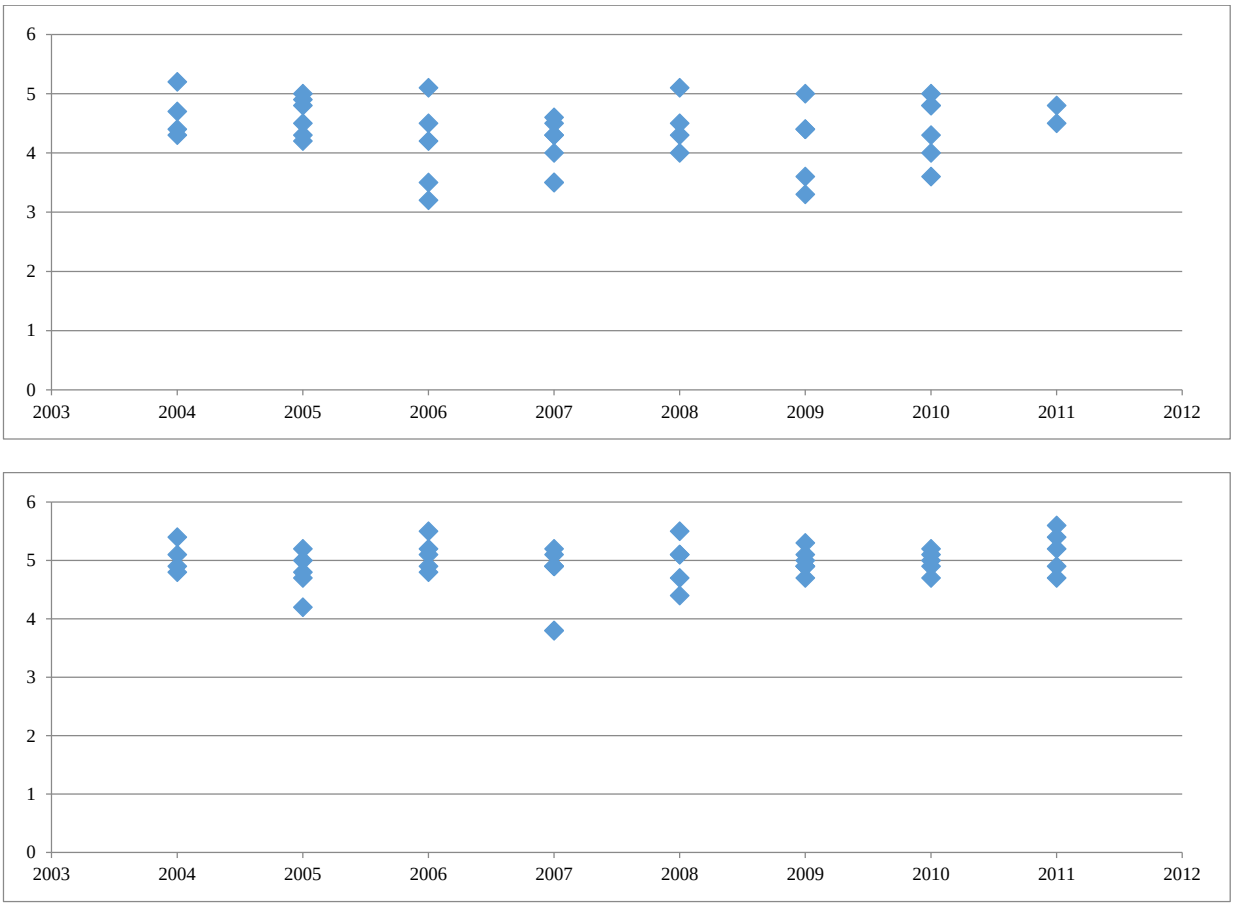

Figure 4.2.2: Two examples of all end of semester ratings for two faculty members at Utah State University (highest possible rating is 6).

Peer teachers largely gave the SGID. In one study, faculty were given a nominal $75 to participate in the SGID [32]. Once the SGID had been completed, the teaching faculty felt better about their teaching and more comfortable having someone observe their class. SGIDs are especially effective for new teachers [32].

If this approach were simplified and efficiently administered, the results would prove valuable. According to several universities who implement this manner of evaluation, these sessions are on average 30 minutes, depending on the size of the class. An added benefit of the SGIDs is that the instructor discusses the results with the exam

administrator who can help to interpret the results.

An alternative to the SGID is another, more recent, idea called the Quick Course Diagnosis (also known as QCD). The QCD builds on the strengths of the SGID. A QCD works like this:

- An instructor informs the students what will happen for the next 15 minutes and leaves while a peer conducts the QCD.
- Students are given an index card and told to give the class a satisfaction ranking and put down one word or phrase that describes why they gave that ranking.
- The facilitator then gives students the learning outcomes for the course and on the reverse side of the index card, students indicate the two learning outcomes the course met and the two that the course least fulfilled.
- The facilitator then breaks the students into groups of 5-7 and who then pass around a sheet of paper, adding ideas about the class on this piece of paper (this is a brainstorming technique called “Roundtable”).
- The group ranks the top 3 strengths and the top 3 weaknesses. The facilitator then records them into a single template and the department analyzes them.
- The department then summarizes them in a histogram and graphs for the benefit of the instructor.

In summary, the QCD differs from the SGID in the following ways:

- QCD captures more data
- QCD requires less time
- QCD offers less possibility for errors
- QCD allows for all ideas to be heard (“Not washed out because of consensus [46]”)
- QCD can be used in large courses

- QCD is more versatile than the SGID: can be used for any academically relevant assessment.
- Administering is “fool-proof” because the quality is from the written products, not the skills of the facilitators.
- QCD allows the generation of reports better than SGID (SGID would require a write-up, whereas QCD is more prone to quantitative data)
- The results are easier to assess and track than an SGID letter.

Because this is more quantitatively based, we can graphically display the results, as in Figure 4.2.3.

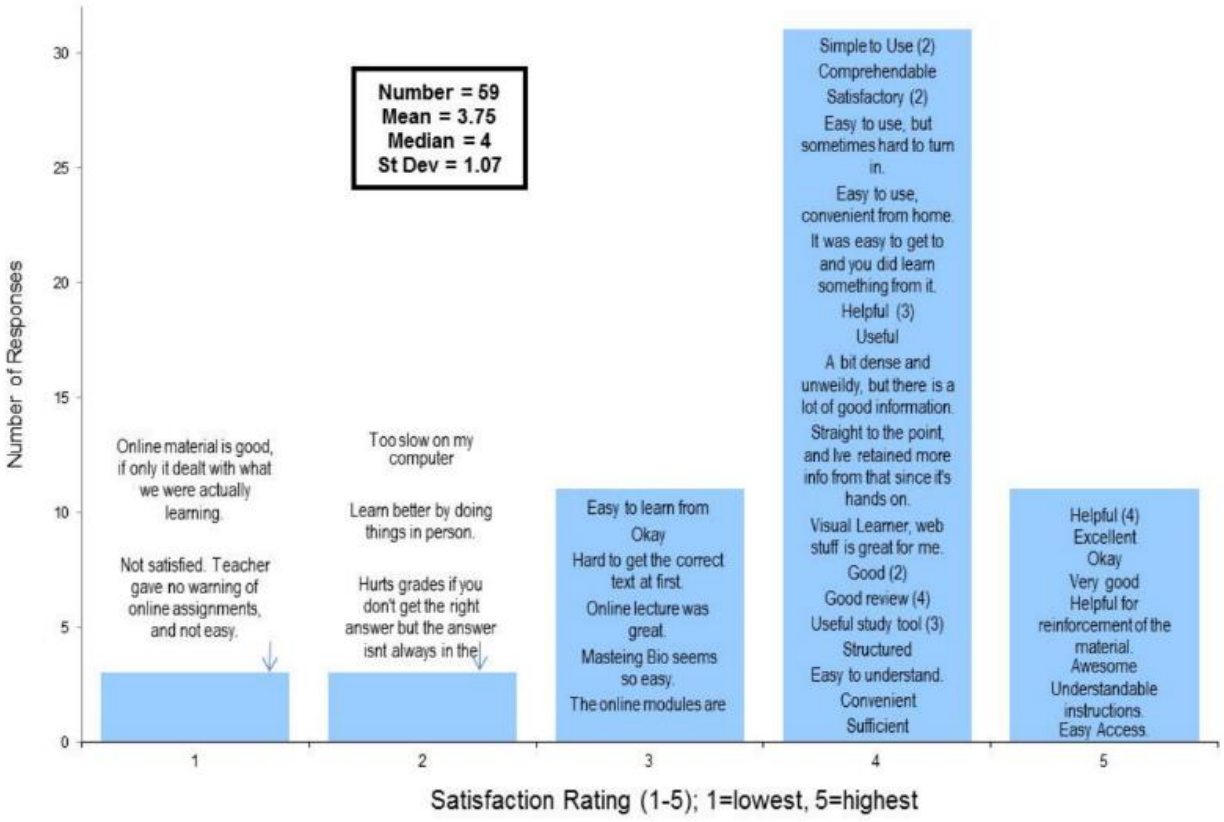

Figure 4.2.3: Example of quick course diagnosis data. Used with permission.

Another suggestion is using Piazza for class evaluation. Piazza is an online collaboration place where students can respond to questions by creating a group edited answer to a question. Currently Piazza can be linked to the Canvas learning management system used by USU classes. Throughout the semester, students can use this interface to ask and answer questions. Instructors can approve student –generated answers or add their own. There are no long threads of answers to read, but an incrementally improving group answer using wiki-like editing. We recommend asking questions relating to classroom workload and practices using the Piazza interface. This way a group answer will be generated, rather than a series of possibly conflicting answers. When students can read the feedback of others, more thoughtful answers will be generated – much like what happens with SGID, but without taking class time.

If the instructors understand the level of their students at different times in the semester, the instruction value could be improved incrementally.

We suggest the following ways to improve the teaching styles of the instructors:

a. Encourage the professors to spend time working homework problems during class lectures or using other methods to create a more hands-on approach to class.
b. Implement a small group instructional diagnosis, quick course diagnosis, or have students input evaluations into Piazza during the middle of the semester. Give the results to the instructor and allow the instructor to make changes appropriate for the individual group of students.

4.2.2 Learning Styles

As part of our research, we considered the learning styles of the students, seniors, and professionals to see if there were differences, or similarities and compared those results to the learning styles of teachers. Felder and Spurlin [11, 12] have studied

students' learning by categorizing their types of learning into four different categories:

1. Visual or Verbal – whether students remember better verbally or visually.
2. Sensory or Intuitive – whether students base decisions on inference or observation
3. Active or Reflective – whether students learn best by talking through the problem or thinking through the problem.
4. Global or Sequential – whether students would rather learn by first seeing the entire perspective or in a logical, stepwise path.

Many studies have been conducted at different universities to categorize students in these four categories. In a survey of engineering students, they found that 64% were active learners, 63% were sensing learners, 82% were visual learners and 60% sequential learners [33].

Sequential learners tend to understand better when given a set of logical steps, while the global learner understands material almost randomly and then suddenly forms a whole picture. Verbal learners get more out of a lecture and talking while visual learners depend more on slides with pictures, charts, figures, and color. Active learners do best when moving around and trying out what they are learning, while reflective learners would rather sit quietly and think about what they have learned. It would be more difficult for an active student to learn while listening to a lecture. Intuitive learners tend toward rapid work and innovation, while sensors are more practical and deliberate.

It is believed that students respond best when taught in their own learning style [33]. When the learning styles and teaching styles of students and teachers are out of alignment, there are repercussions on the students' ability to learn.

We do not know whether learning styles change as people grow and mature, and are thus faced with different learning situations. Because of the advent of handheld technology (such as smartphones) and increasingly interactive coursework (such as

group learning), active learning is more familiar to younger students. Staying focused on hour-long lectures becomes more difficult for students who are always carrying these interactive devices and continually connected to social life. Because of this, we look at the learning styles of different populations. For an understanding of how this information correlates to teachers' learning styles, we combined our learning styles data with that of Felder and Brent [33]. In the Felder paper, they considered a person to be either Active or Reflective (not allowing for a Neutral designation). To compare with their findings, we recategorized our results so that each scale was interpreted in a binary fashion (e.g., Active or Reflective). Table 4.2.3 indicates interesting differences between various groups of individuals.

While all four groups appear to be visual learners, we see a greater tendency to be Active and Sequential among the younger respondents. We need to engage students. These results show that, in two of the four scales, the engineering faculty tend to be less sensing, active and sequential than the students they are teaching. In choosing a teaching style, faculty are influenced by both the way they were taught and the way they like to learn. This could result in a mismatch between how teachers teach and how students like to learn. One of our students vocalized a common theme throughout our surveys, "I have a hard time following his style of teaching. I'm not sure how he could teach it better, but maybe present it in a more interesting manner." Another student said this about the teaching style of the professor, "[I wish I had found] a different instructor that employs different teaching methods and is more approachable." This speaks to the need for instructors to learn how to teach in a manner the students will understand.

If we look at the data for the high school students, and even apply a stricter standard of what it means to be Active or Visual (on a scale of -11 to 11, we counted only those at the extremes (5,11) or (-11,-5) as possessing the attribute), 85% are either active or

visual. For the upper division Computer Science students, 73% are either active or visual. Thus, Visual/Active activities are a good match for the majority of students. If an instructor ignores this and focuses merely on traditional lecture, the course will likely be less effective.

Table 4.2.3: Percent of students, professionals and faculty which fall into the various learning categories. *Data taken from [33]

	Active	Sensing	Visual	Sequential
Math/Physics High School (n=128)	73	53	88	65
Upper Division CS (n=86)	55	59	90	55
CS Professionals (n=42)	40	57	83	45
Engineering Faculty* (n=101)	45	41	94	44

Our recommendation for improvement in this area would be to simply create a more hands-on learning environment.

4.2.3 Creating a Pre-Introductory Course

The attrition problem has been well documented in this paper. At a recent ACM SIGCSE conference, speakers reported that women CS majors were often surprised by the amount of creativity there was in later courses because introductory courses did not highlight this part of computer science [34]. Women, more than men, tend to have interest in the real application of computing instead of just computing for computing's sake [35, 36].

An additional problem in computer science is the underrepresentation of African Americans and Hispanics among computer science majors. This problem has been studied less than the shortage of women possibly because there are so few computer science majors from these groups that it is difficult to make any statistically meaningful statements about their numbers.

A recent study at the University of Illinois at Chicago detailed the start of and changes to a pre-introductory course entitled CS 0.5. After seeing the success of this course, it was implemented as mandatory to either take or test out. Initially this course was a basic web development course discussing HTML and JavaScript. This was successful, but was found to not prepare students for the coming computer science courses. This was then changed to learning computer science principles using the Python language. They decided to teach Python instead of Java so that only the principles were discussed and the language was given less importance, as they teach Java in later courses [8]. The difference between the old way of teaching CS 0.5 and this newer way with Python resulted in much higher rates of retention and success (success being defined as obtaining an A, B, or C grade at the end of the semester). Success increased from 75.9% to 84.1% and the retention rate also increased from 38.1% to 59.1% (see Table 8). The authors note that, all things being equal, the success rate should have *decreased* because starting in the spring of 2005, a greater effort was made to remove the best students from the course and place them in CS 1.

Table 4.2.4: Success and Retention Rates at the University of Illinois at Chicago using an old and new approach in CS 0.5. Retention is a percentage of students who enrolled in CS 202, which is the third course after CS 0.5.

UIC's CS 0.5	Enrollment	Success	Retention
Fall 2002	61	74.80%	42.10%
Spring 2003	38	76.70%	22.20%
Fall 2003	51	68.60%	38.60%
Spring 2004	22	82.90%	44.40%
Fall 2004	15	93.30%	44.40%
Average "Old"	**37**	**75.90%**	**38.10%**
New Spring 2005	18	94.40%	62.50%
New Fall 2005	29	90.00%	57.10%
New Fall 2006	42	76.20%	63.90%
New Spring 2007	24	83.30%	59.10%
Average "New"	**28.3**	**84.10%**	**59.10%**

Another similar article was published detailing the need for students to be given more than simply programming to work on during the introductory course because more and more universities are requiring the introductory course to computer science be taken by non-majors. Because of this, educators at Georgia Tech decided to make what was being learned more relevant to a wider base of students by introducing media creation into the course. Students were instructed to use their own photographs and other media and use computers to manipulate them into something they want. The drop, withdrawal, D or F rates dropped more than 30% because of this change in the introductory program [37].

We recommend that programs look at whether a pre-introductory course would be beneficial to their computer science program and implement them as needed.

4.2.4 Adjusting Math Requirements

Professionals and seniors mentioned that there is too much math. The following is stated as mathematics requirements for ABET accreditation [38]:

Mathematics: At least one half year that must include discrete mathematics. The additional mathematics might consist of courses in areas such as calculus, linear algebra, numerical methods, probability, statistics, number theory, geometry, or symbolic logic.

We examined computer science pre-requisites in 10 schools' computer science course listings as shown in Table 4.2.5. Most courses do not have any mathematics requirements beyond college algebra, which means that this advanced math, which is a deterrent for many to join computer science, may be unnecessary for most of the computer science courses.

In order to validate this, we also took the computer science major requirements of these ten universities (not just their course listings) to see whether the math required for computer science is actually used in the computer science core courses (see Table 4.2.5). What we found is that in most computer science programs, the highest math that is actually listed as a prerequisite for a core course in most of the programs (aside from discrete mathematics) is calculus I. Only one university from the list requires calculus II as a prerequisite for any of its required courses. Five out of the ten universities had pre-requisites less than calculus. The University of New Mexico has no specific math course required, but does have a foundational math course specific for computer science majors that has a CS prefix (meaning, it is not taught by the math department, but is specific for computer science).

With all of these programs, however, we find that calculus II is required to graduate in the computer science program, even though this level of math, in most cases, is not even used. For some schools, discrete math (required by ABET) has a prerequisite of calc II.

However, in other schools, discrete math only requires pre-calculus. Thus, by introducing a discrete math class with reduced prerequisites, math requirements could be lessened without affecting ABET accreditation.

Table 4.2.5: Math requirements listed as pre-requisites for core courses

School	Highest Math Listed as a prerequisite	Highest Math listed as a program requirement
Boise State University	Pre-Calculus	Calc II
Central Connecticut State	Calc I	Calc II
Central Florida	Trigonometry	Calc II
Colorado University	Calc I	Linear Equations and Calc II
Idaho State University	College Algebra	Linear Algebra and Calc II
Montana State University	Calc II	Linear Algebra and Calc II
Southern Utah University	College Algebra	Linear Algebra and Calc II
University of Nevada Las Vegas	Calc I	Linear Algebra and Calc II
University of New Mexico	CS foundational Math	Linear Algebra and Calc II
University of Oklahoma	Calc I	Linear Algebra and Calc II
Utah State University	College Algebra	Calc II

When we asked an open-ended question “Why don’t more students pick your field of expertise?” to the professionals, 22% responded that there is too much math involved. This was the second highest response (see Table 4.2.6).

Table 4.2.6: (Open-ended) Why professionals think students do not choose their major (N=70)

In your opinion, why don't more students pick your field of expertise?	percentage
Hard	42%
Too much math	22%
Stereotypes	12%
Busywork in introductory courses/ irrelevant courses	8%
Outsourcing of jobs	6%
Modern Students Have No work ethic	4%
Irrelevant to business	2%
Counseled against it	2%
Boring	2%

These data indicate that the level of math is a deterrent and effort should be spent checking whether a certain math class is really necessary for a student, considering the emphasis of the student. A Vision emphasis may need the heavy calculus required, but another emphasis, such as software development, may not need it. One professional mentioned that students and teachers “need to understand that in most jobs, you'll just be doing algebra, and simple math equations.” Is it necessary for each student to have passed calculus II in order to be a successful programmer? Granted, there are other tracks that would definitely require a deeper understanding of mathematical problems, but for many students, this level of mathematical understanding may not be necessary. Math may help with the logical thinking ability [39, 40, 41] that is necessary in computer science, but it is not clear that computer science itself cannot provide such training. Perhaps a good study for future work would be one which investigates the degree to which graduates have felt math gave them needed depth or thinking ability.

4.2.5 Creating Clear Assignments

We asked the first semester students the cause of their frustrations. The percentage of students who mentioned confusing assignments was more than 60% (see Table 4.2.7). These data show this could be a major stumbling block for many students, especially those just starting out in computer science. This could mean there is a disconnect between teaching and learning styles. This could also mean that assignments are not covering what is being discussed in class. One suggestion to address confusing assignments is to create a chat room for peers to help each other with assignments. Piazza is ideal for this as the user interface is well developed. Students can use their smart phone to upload a diagram of their solution. Students could be given credit for their participation. Tutors or teaching assistants could routinely monitor the status of questions and answers. Usage statistics could be used to inform instructors of the difficulties created by assignments.

We asked the first semester students an open-ended question describing their favorite computer science assignment and why it was their favorite. When coupled with the reason students choose a major in the first place (finding a job after graduation), creating assignments that students will be able to show a prospective employer have an added bonus.

Table 4.2.7: First semester students frustrations in class (N=53)

My frustrations in class have the following causes (select all that apply)	
Answer Options	Response Percent
Assignments are confusing. I often have no idea how to begin.	60.4%
The instructor assumes I have previous knowledge I don't have.	26.4%
Exams aren't representative of my skills	22.6%
Assignments are too time consuming	20.8%
Assignments are tedious	20.8%
Class is boring	15.1%
The textbook is difficult to read	15.1%
My grades do not reflect the time I spend on the class	11.3%

When asked whether students have shown programs which they wrote to family or friends, eighty percent of the students replied in the affirmative (see Figure 4.2.4). Increasing the “showability” of assignments also increases advertisement for the major.

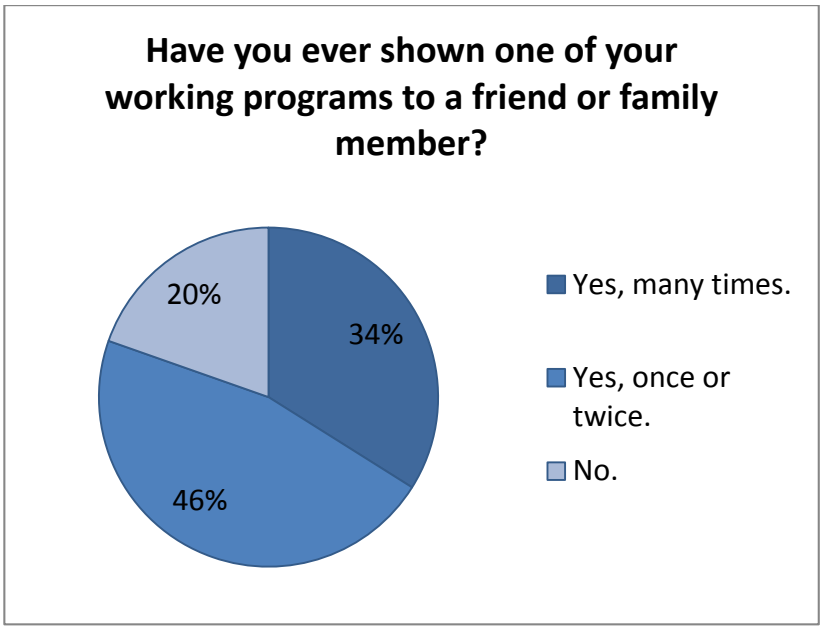

Figure 4.2.4: First semester students showing work to others (N=56).

When asked about the least desirable things in the major, 13.3% of the senior students responded with some answer related to dead-end courses or assignments. Students want assignments that are meaningful. One student described the assignments as "busy work." Another said, "[This is] course work that will never apply to anything in my future career." Whether that last statement is true or not does not matter as much as whether the student *thinks* that it's true. If the students do not understand the value of what they are doing, both the satisfaction and amount of effort spent on the assignment will likely decrease.

One student lamented, "[There were] oversimplified problems in class - HORRIBLY difficult problems on assignments and tests." Another student said, "[There were] unclear assignments where the teachers ask for every detail and you can't figure out what in the world they are saying." These last two statements highlight the communication and learning style problem that we've discussed earlier as well as the assignment clarity

problem we are discussing currently. We cannot be expected to hold the students' hands while they are programming every small piece, as it seems many students want. We can, however, be more available and offer more resources for the students to learn what they need to be successful. Many assignments are too specific in what is to be done and how, so that creativity is stifled. The exact specification makes compliance confusing.

While these suggestions are related to teaching styles of specific instructors and experiences of individual students, a case can also be made for an overall change in how an assignment is created. A group of professors at North Carolina State University were curious about how to create a meaningful assignment (one with social relevancy), and whether meaningful programs were being created as learning tools for computer science students. They found that only 34% of CS1 projects had a practical or socially relevant context, 41% had no context at all and 15% were games [13]. They pointed out that Millennial students (those born after 1982) are in search of a career with meaning [42]. We in the computer science community understand the social importance of computer science. We understand that the skills they learn while programming robots could help cure cancer or put an astronaut into space, but can the students see these noble applications? Are students leaving computer science in search of a major that has a better chance of giving them a career with a more socially-relevant purpose?

We suggest the following ways to improve the appeal of assignments:

a. Have a few students and the teaching assistant evaluate an assignment for readability before it is finalized.
b. Have current students suggest assignments for the next semester. Give bonus points (or other incentives) for assignments that are selected.
c. Evaluate each assignment based on the following criteria: Would a student be interested in the results? Would a person want to show the working program to

friends? Do the details stifle creativity? Does the assignment make clear the application to important real-world problems?

d. Once an assignment has been given, post a current list of questions (and answers) about the assignment.

We will leave as future work a more extensive research project on what makes a good assignment.

4.2.6 Creating a Better Learning Environment

63% of first semester students we surveyed indicated only moderate experience using computers. Knowing this level of experience will help teachers connect better with their students, and therefore improve their teaching. While students who have been programming a long time exist, this shows that they are the exception, rather than the rule.

How does a teacher create a better learning environment? One way is to reinforce what is being taught in the classroom by creating out-of-classroom assignments that are interesting. 67% of students we surveyed said that using Interactive Computerized Learning Modules, such as those found at csilm.usu.edu are their first choice for method reinforcement (see Figure 4.2.5).

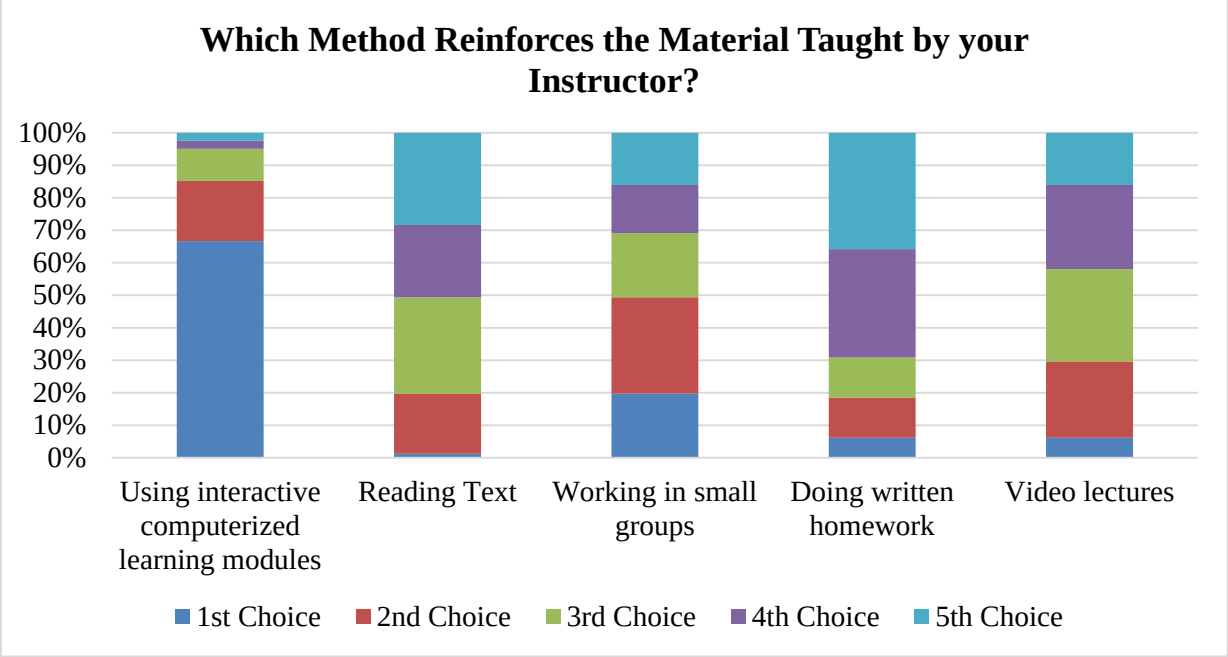

Figure 4.2.5: Students' preferred method of reinforcing material taught by instructors (N=81).

These modules reinforce difficult concepts such as recursion and loops. Using these modules could prove valuable in helping a beginner student come to the level of the more advanced students.

Students also rated working in small groups high (see Figure 4.2.5). Instructors would do well to create a category of assignments that encourage collaboration in small groups.

Seventy-five percent of students we surveyed indicated they want feedback on whether the final solution is correct or not and 52% want periodic feedback about sub problems. Giving students timely and appropriate feedback on assignments is crucial to the students' improvement; it is also something they want.

Sixty-eight percent of students we surveyed understand better when they "try it out"

versus “think it through.” Sixty percent of seniors we surveyed feel they only have a moderate understanding of the material. We can increase understanding of students by using Interactive Learning Modules and more hands-on activities. Sixty-eight percent of seniors we surveyed said instructors used, “few, if any, activities” to teach their students. The current course of action hampers students’ ability to not only learn, but retain their knowledge. While we have no firm evidence to show that slides really do kill learning, using and evaluating a variety of approaches to teaching is important.

Seniors generally liked the assignments they were given, but the lectures were seen as dull and with little student interaction (see Figure 4.2.6). Focusing on the assignments during the class provide both clarity about the assignment and more opportunities for hands-on learning. Students who are actively engaged in thinking and posing questions are more likely to retain important concepts.

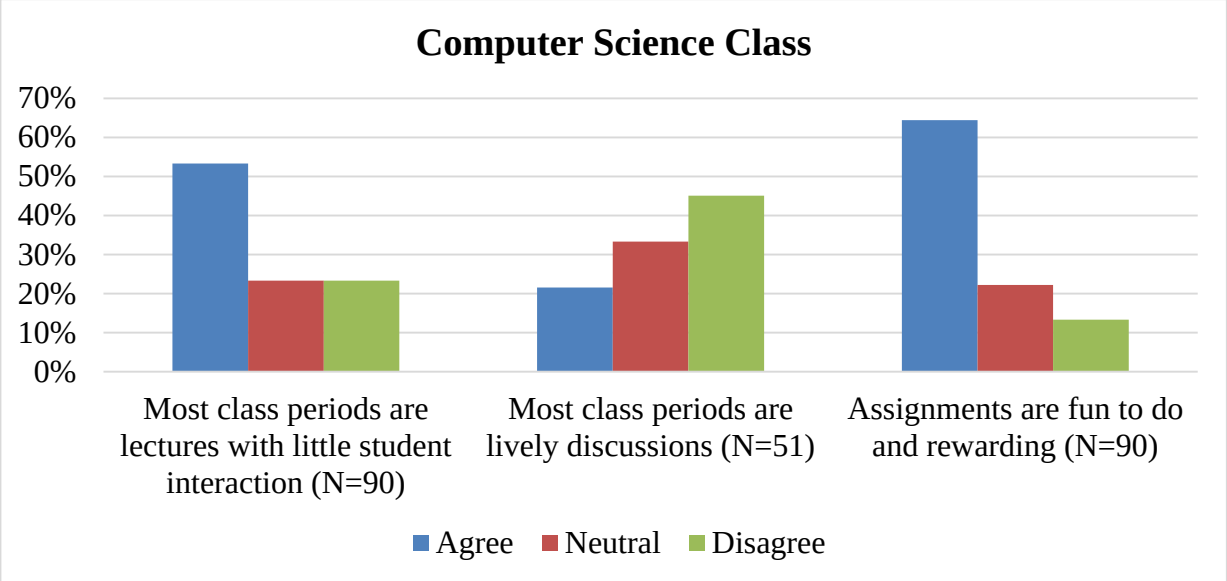

Figure 4.2.6: Students' feelings on the way computer science is taught.

Reviewing seniors' comments on creating better learning environments helps us understand this principle:

- "More hands on + lab type work in initial years. Expand on the CS1400 labs. Should cement understanding; making them more capable of understanding later material."
- "More real-world examples. [Giving] Understanding to those that don't think, eat, and breathe computers and math."
- "More demonstrations. Some teachers do this quite well, but there are several that do not use this avenue of teaching."
- "More hands on activities and less lectures."

It is clear from the data that students want more interaction in lectures, more practice programming, and more activities to cement understanding. We recommend that teachers utilize interactive learning modules and small-group assignments.

In our CS3 classes, we used online quizzes to help direct student learning, give feedback, encourage group interaction, and prepare students for exams. These online quizzes were administered by our Learning Management System, which allowed immediately feedback (as quizzes were automatically graded) and retries. Students were encouraged to work on the quizzes in small, dynamically created groups. Written homework has long been a part of this course, but converting the written homework to the repeatable quiz format was successful. Figure 4.2.7 shows that students were very positive about both the interactive learning modules and the online assessment.

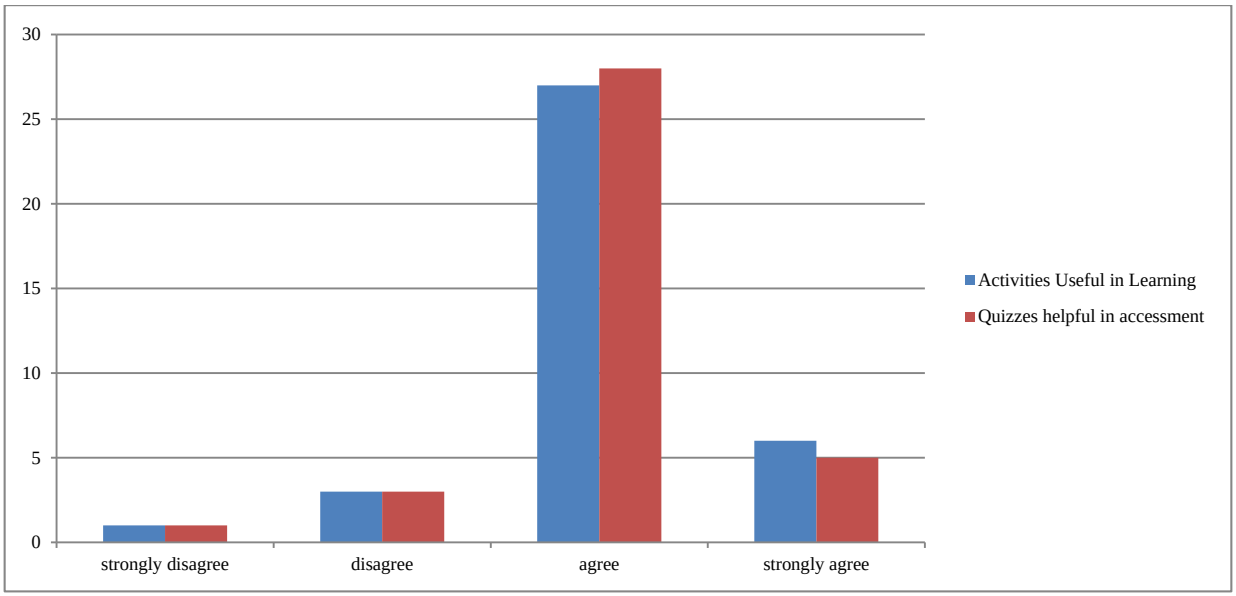

Figure 4.2.7: Students' opinions on learning modules and quizzes.

It is important to understand the things that are least desirable in computer science from an insider's perspective in order to create a better learning environment. We asked an open-ended question mentioned earlier "What are the least desirable things about your major?" The responses were varied, but there were many which were similar. Table 4.2.8 shows these results.

Table 4.2.8: (Open-ended) Least Desirable Things about Major from a Senior Perspective (N=84)

What are the least desirable things about your major?	Count	Percentage
Takes a lot of time/work	45	37.2%
Math	15	12.4%
Meaningless courses/assignments	11	9.1%
Professors' teaching styles	9	7.4%
Steep learning curve	8	6.6%
Male-Female ratio	6	5.0%
Dealing with Stigma	6	5.0%

As we can see from the Table 4.2.8, 37.2% of students indicate that the major takes a lot of time, which is the highest of the least desirable things in computer science. Brown suggests providing students with an iterative and incremental process in assigning workload [43]. For example, use started code for an assignment which increases a particular skill instead of a program that the student would need to complete from scratch. We suggest monitoring the amount of time students report spending on assignments to help both students and teachers evaluate time use. Giving starter code is a good way to allow students to get targeted experience without requiring as much time. Systems such as csilm.usu.edu offer alternatives to programming through interactive learning modules (ILMs). For example, the Loops ILM (shown in Figure 4.2.8) provides experience with nested loops in an engaging and challenging setting. Students control loop variables, increments, and tests to match specific goal patterns. Such activities would also help the students have a more hands-on learning experience, which in Figure 4.2.7 is discussed as a suggestion for improving the education students receive.

Another idea that has recently gained traction with the advent of more mobile and internet capable devices is a flipped classroom. This approach was developed at The Ohio State University as an alternative to the traditional lecture in class and homework outside of class [44]. The flipped classroom gives students a recorded lecture to watch outside of class. Traditional classroom lecture time is utilized to do homework and hands-on activities with the professor there to guide learning.

We queried seniors on ways to improve our computer science program. They overwhelmingly agreed that we need to improve the quality of instruction and use more variety in teaching methods (see Figure 4.2.9).

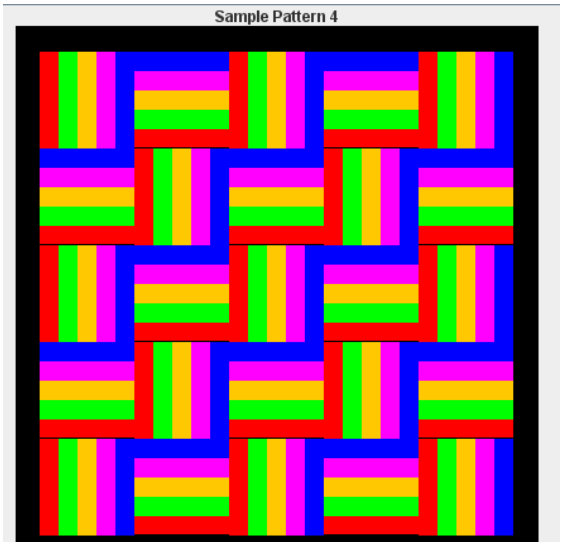

Figure 4.2.8: CSILM.usu.edu loop applet allows students to practice nested loops in creating attractive patterns.

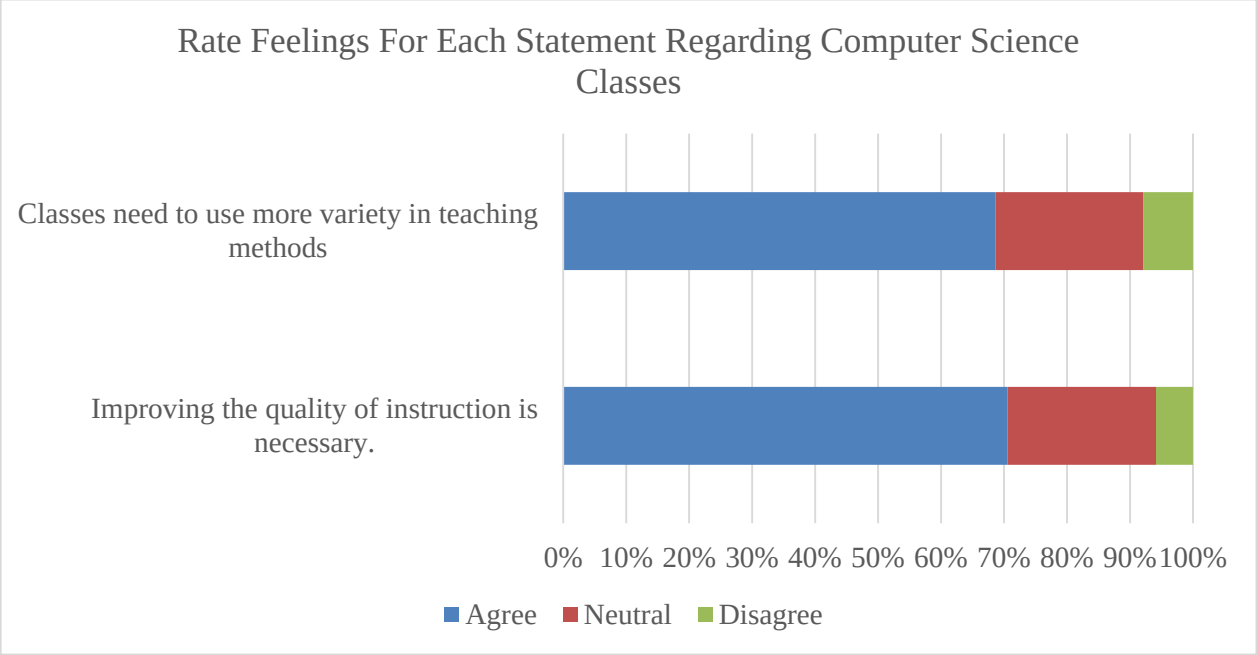

Figure 4.2.9: Rating improvement statements from a senior perspective (N=51).

We recommend that computer science departments look at their individual programs to see whether any of the following would be helpful in improving computer science programs:

- The Flipped Classroom approach developed by The Ohio State University
- Interactive Learning Modules such as those found at csilm.usu.edu to supplement teaching and provide learning activities
- Suggesting professors to use more in-class coding and interactive teaching methods

CHAPTER 5

CONCLUSIONS

We have demonstrated throughout this thesis many reasons to increase enrollment in computer science in the United States. We have also given researched recommendations for improving enrollment. Each individual program will need to look at their specific needs and implement the recommendations each program sees necessary. We are not advocating every recommendation be implemented in every university and college throughout the country, but these suggestions should provide a solid framework for discussion.

One major contribution of this thesis is the collection of surveys completed between the seniors, professionals, and first semester students. Other major contributions include researched recommendations for improving computer science programs; combining other related works with our own study. The qualitative comments as well as the quantitative data from each of these groups provide a computer science life cycle that has not been approached in a single study. Based on this unique approach, we found the following:

- Current recruitment efforts are insufficient. Based on feedback from our studies, we found that students need to be recruited earlier, and potential recruits need a better understanding of what computer science is.
- Many current mathematics requirements for computer science are unnecessary and should be looked at based on the student's emphasis.
- Current recruitment and retention efforts for women in computer science are ineffective and computer science programs need to create a more female-friendly atmosphere by creating courses designed for female strengths and interests.

- Teaching and learning styles of teachers and students may not match. Current evaluation systems are insufficient in giving students the tools to convey when this mismatch is happening, such as SGIDs, QCDs, or the Piazza system.
- Many students coming into computer science are ill-prepared. We have proposed a pre-introductory course as a tool for those students

Computer science courses could be more effective in creating a hands-on learning environment. Based on our research, we proposed students be given clearer assignments, and courses use Interactive Learning Modules or upside down courses to accomplish this necessary change.

The obvious next step in this research is to implement these ideas in a controlled setting and see which are effective and which are not. While theorizations and ideas are important, these often differ from real implementation. Some may be easier to implement than others. We recommend studying those ideas that are the easiest to implement and move forward with more difficult ideas.

REFERENCES

[1] A. C. Sherry et al., “Assessing distance learners' satisfaction with instruction: A quantitative and a qualitative measure”, *American Journal of Distance Education*, vol. 12, no. 3, pp. 6-9

[2] A. P. Carnevale et al., *STEM Science, Technology, Engineering, Mathematics*, Georgetown University Press, 2011, pp. 20-29

[3] S. Zweben, “Computing Degree and Enrollment Trends,” Computing Research Association, Washington, DC., 2012

[4] Bureau of Labor Statistics. (2013, September 15). *Employment Projections 2010-2020*, [Online]. Available: http://www.bls.gov/emp/

[5] National Center for Education Statistics (2013, October 15). *Digest of Education Statistics* [Online]. Available: http://nces.ed.gov/programs/digest/d11/

[6] H. Salzman et al., “Guestworkers in the high-skill U.S. labor market”, Economic Policy Institute, Washington, DC., Rep. 359, Apr. 2013

[7] OECD, “Moving Up the Value Chain: Staying Competitive in the Global Economy”, OECD.org, Paris, France, 2007

[8] R. H. Sloan and P. Troy, “CS 0.5: A Better Approach to Introductory Computer Science for Majors” in *Special Interest Group on Computer Science Education*, Atlanta, GA, 2008, pp. 1-5

[9] K. Rask, “Attrition in STEM Fields at a Liberal Arts College” in *Cornell University School of Industrial Relations*, Ithaca, NY, 2010, pp. 1-18

[10] R. M. Powell, “Improving the Persistence of First-Year Undergraduate Women in Computer Science” in *Special Interest Group on Computer Science Education*, Atlanta, GA, 2008, pp. 1-5

[11] R.M. Felder and B.A. Soloman, "Systems Thinking: An Experimental Course for College Freshmen," *Innovative Higher Education*, vol. 19, no. 1, pp. 3-17, 1988

[12] R.M. Felder and J. Spurlin, "Applications, Reliability and Validity of the Index of Learning Styles"*, International Journal of Engineering Education*, Vol. 21, pp. 103-112, 2005

[13] M. A. Chandler, "Fewer high school students taking computer science classes", *The Washington Post*, Dec. 2009

[14] N. WingField, "Fostering Tech Talent in Schools", *The New York Times*, Sep. 2012

[15] J. Luczaj, "Creating a Summer Program to Engage Students", *Journal of Computing Sciences in Colleges*, vol. 25, no. 5, pp. 319-325, May 2010

[16] M. Sabin et al., "Designing and Running a Pre-College Computing Course", *Journal of Computing Sciences in Colleges*, vol. 20, no. 5, pp. 176-187, May 2005

[17] L. Carter, "Why Students with an Apparent Aptitude for Computer Science Don't Choose to Major in Computer Science", *Special Interest Group on Computer Science Education*, vol. 38, no. 1, pp. 27-31, March 2006

[18] K. Wagstaff, "Can We Fix Computer Science Education In America?"*, Time Magazine*, July 2012

[19] The College Board, "AP report to the nation 2012," *The College Board*, Feb. 2012

[20] A. Tucker et al., "A model curriculum for K–12 computer science," Association for Computing Machinery*,* New York, NY, Rep. 104043, 2003

[21] National Research Council Committee, *Being fluent with information technology*, Washington, DC: National Academy Press, 1999

[22] E. Roberts and G Halopoff, "Computer Science Teachers Association analysis of high school survey data", unpublished.

[23] C. Stephenson, "The New Educational Imperative: Improving High School Computer

Science Education", unpublished.

[24] R. Hammond, "Overcoming Geek Mythology: Computer Science Opens Its Doors to Women", *Carnegie Mellon Magazine*, vol. 19, no. 3, pp. 13-17, 2001

[25] E. Spertus, "Why Are There So Few Female Computer Scientists?," *Massachusetts Institute of Technology*, Cambridge, MA, Rep. 1315, Aug. 1991

[26] M. Johnston, "Women in I.T.: Looking Beyond the "Girl Geek" stereotype," *Networker Magazine,* vol. 5, no. 3, pp. 40 - ff, 2001

[27] S. Graham and C. Latulipe, "CS Girls Rock: Sparking Interest in Computer Science and Debunking the Stereotypes," *ACM SIGCSE Bulletin*, vol. 35, no. 1, pp. 322-326, Jan. 2003

[28] J. Mason and T. Beaubouef, "Why the High Attrition Rate for Computer Science Students: Some Thoughts and Observations," *ACM SIGCSE* Bulletin, vol. 37, no. 2, pp. 103-106, Jun. 2005

[29] C. Hill et al., *Why So Few? Women in Science, Technology, Engineering, and Mathematics*, Washington, DC.: American Association of University Women, 2010

[30] J. Hattie and H. W. Marsh, "The Relationship Between Research and Teaching: A Meta-Analysis," *Review of Educational Research*, vol. 66, no. 4, pp. 507-542, 1996

[31] B. Ischinger et al., *Creating Effective Teaching and Learning Environments: First Results from TALIS*, Paris, France: OECD Publications, 2009

[32] J. A. Sanchez and T. E. Margett, "Small Group Instructional Diagnosis (SGID) Fall 2011 Semester Report," *The University of New Mexico*, Albuquerque, NM, 2011

[33] R.M. Felder and R. Brent, "Understanding Student Differences", *Journal of Engineering Education*, Vol. 94, No. 1, pp. 57–72, 2005

[34] S. L. Peeger et al., "Increasing the enrollment of women in computer science." In *Proc. Thirty-second SIGCSE Technical Symp. on Computer Science Education*, 2001, pp.

386–387

[35] C. Hill et al., *Tech-Savvy: Educating Girls in the New Computer Age* New York, NY: American Association of University Women, 2000

[36] J. Margolis and A. Fisher, “Unlocking the Clubhouse: Women in Computing,” *ACM SIGCSE Bulletin - Women and Computing*, vol. 34, no. 2, pp. 79-83, Jun. 2002

[37] A. Forte and M. Guzdial, “Computers for communication, not calculation: Media as a motivation and context for learning,” In *HICSS '04 Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, vol. 4, no. 4, pp. 40096.1, 2004

[38] *ABET Criteria for Accrediting Computing Programs*, 2012

[39] S. T. Leung and C. Johnson, “Computational Concepts in IT: A New Approach to IT Mathematics” in *SIGITE '05 Proceedings of the 6th conference on Information technology education*, pp. 37-42, 2005

[40] P. Petocz and A. Reid, “The Contribution of Mathematics to Graduates’ Professional Working Life” in *Australian Association for Research in Education*, vol. 5, no. 1, pp. 1-11, 2005

[41] F. Marton and S. Booth, *Learning and Awareness,* Lawrence Erlbaum, Mahwah, NJ: Lawrence Erlbaum Associates Inc., 1997

[42] L. Layman et al., “Note to Self: Make Assignments Meaningful” in *ACM SIGCSE Bulletin*, vol. 39, no. 1, pp. 459-463, Mar. 2007

[43] J. Brown, "Bloodshot eyes: workload issues in computer science project courses," *Software Engineering Conference, 2000. APSEC 2000. Proceedings. Seventh Asia-Pacific* , vol., no., pp.46,52, 2000

[44] J. Strayer, “The effects of the classroom flip on the learning environment: a comparison of learning activity in a traditional classroom and a flip classroom that used an

intelligent tutoring system," Ph.D. dissertation, Phil. Dept., Ohio St. Univ., Columbus, OH, 2007

APPENDICES

APPENDIX A: First Semester Survey 1

1. **Clicking on the "agree" button below indicates that: • you have ready the above information • you voluntarily agree to participate • you are at least 18 years of age If you do not wish to participate in the research study, please decline participation by clicking on the "disagree" button.**
    - Agree
    - Disagree
2. **What is your student number?**
3. **What is your email address (this will be used to send you future survey links and to arrange for mailing of the incentive)?**
4. **What class are you attending?**
    - Colorado State University CS160 (Whitley)
    - Colorado State University CS160 (Wakefield)
    - Colorado State University CS160 (Wilcox)
    - Utah State University CS1 (Watson)
    - Utah State University CS1 (Cooley)
    - Brigham Young University - Idaho (CS 165)
    - Brigham Young University - Idaho (CS 124)
    - Southern Utah University (CSIS 1400)
5. **What is your gender?**
    - Female
    - Male
6. **How old are you?**
7. **Thinking about your life outside coursework (example: Family, Work,**

etc.), how would you rate your stress level in these areas? Would you say your life is...

- Not stressful at all
- Slightly of stressful
- Stressful
- Extremely Stressful

8. **Have you decided on a major? If so, what?**
9. **What best describes your reason for registering for a computer class?**
    - Course is required for my major
    - Course is required for my minor
    - Course fills general requirements
    - Course is not required, but I am interested in the subject
    - I needed any class and this one was available.
10. **When I think about finishing a college degree**
    - It is certainly a reachable goal
    - It would take too long to be realistic for me
    - Explain if necessary
11. **Please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral;somewhat disagree; strongly disagree)**
    - I am excited about the current semester
    - I am looking forward to the challenge ahead of me
    - I will be able to handle all the assignments and get a good grade
    - I'm interested in seeing how hard computer science is

- I am nervous about my ability to succeed

12. **Which of the following activities have you participated in? (check all that apply)**

    - General Computer Applications (e.g., MS Works, MS Office, Open Office)
    - Word Processing (e.g., MS Word, WordPerfect, Open Office)
    - Spreadsheets (e.g., MS Excel, Open Office, IWork)
    - Presentations (e.g., MS Powerpoint, Keynote, Prezi, Open Office)
    - Desktop Publishing (e.g., MS Publisher, InDesign, Pagemaker)
    - Mathematical Applications (e.g., MatLab, Mathcad, Mathematica)
    - Web Design/Development (e.g., Dreamwearver, MS Expression, Web Studio
    - Multimedia Design (e.g., Photoshop, Imovie, Premiere, Movie Maker, Illustrator)
    - Computer Aided Design (e.g., Autocad, Inventor 3D)
    - Game Design (GameMaker, Blender, Maya, Visual Studio/Basic/C++, C#)
    - Mobile application development (Objective C, AppInventor, Iphone/Android, Ipad/Tablet)
    - Web application development (Facebook Api, Flash, Java, HTML5, PHP)
    - Computer Technology (e.g.,install/maintain computers for yourself or software installation)
    - Network/System Operations, Maintenance (e.g., system manager for

multiple machines)

- Network design/development (e.g., LAN, wireless)
- Programming with Robotics (e.g., Labview, ROBOTC, LEGO Mindstorms)
- Programming with Graphics (e.g., Scratch, Alice, Flash, OpenGL, Java 3D)
- Programming language (e.g., Visual Basic, Python, Java, C++)
- Database design/programming (Access, MySQL, Filemaker)
- Other (please specify)

13. **In which of the following extra-curricular computing and information technology activities have you participated?**
    - Computing/Technology-related club or team
    - Computing/Technology-related community involvement/volunteerism
    - Independent study of computing/technology to advance your knowledge
    - Computing/Technology-related job
    - Computing/Technology-related internship
14. **Which of the following classes do you plan to take (or have already taken)? Check all that apply)**
    - AP Computer Science in High School
    - Algebra I
    - Algebra II
    - Pre-calculus/Trigonometry
    - Business Calculus
    - Calculus

- Statistics
- Biology
- Physics
- Chemistry

15. **How many hours per week do you play video/computer games?**
    - none
    - less than 5
    - 5-10
    - 10-15
    - 15-20
    - more than 20
16. **How many hours per week do you participate in social media (such as blogs, facebook, twitter, Google+)?**
    - none
    - less than 5
    - 5-10
    - 10-15
    - 15-20
    - more than 20
17. **How do you categorize your experience in using computers?**
    - No experience
    - Low experience
    - Moderate Experience
    - High Experience

18. **Of the following activities, rank each activity in terms of the method you would like most to use to reinforce the material taught by your instructor. First choice represents your most desirable method of learning, while Fifth choice represents your least desirable method of learning.**
    - Using interactive computerized learning modules
    - Reading Text
    - Working in small groups
    - Doing written homework
    - Video lectures

APPENDIX B: First Semester Survey 2

1. **What is your student number?**
2. **What class are you attending?**
    - Colorado State University CS160 (Whitley)
    - Colorado State University CS160 (Wakefield)
    - Colorado State University CS160 (Wilcox)
    - Utah State University CS1 (DuHadway)
    - Utah State University CS1 (Cooley)
    - Brigham Young University - Idaho (CS 165)
    - Brigham Young University - Idaho (CS 124)
    - Southern Utah University (CSIS 1400)
3. **How many hours per week do you work at a job?**
    - I don't have a job
    - 1-5

- 6-10
- 11-20
- 21-30
- 31-40
- more than 40

4. **Thinking about your life outside coursework (example: Family, Work, etc.), how would you rate your stress level in these areas? Would you say your life is...**
    - Not stressful at all
    - Slightly of stressful
    - Stressful
    - Really Stressful
5. **With respect to the computer science course you are taking, please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**
    - I understand the material presented.
    - I feel like I can contribute to the class
    - I feel supported when I have questions
    - Help is there when I need it
    - I feel like I have friends in the class
    - I have sought outside help frequently to help with programming.
6. **When you need help with class material, where do you go for help? (Never; Once or Twice; Frequently; On an almost daily basis)**
    - My instructor, during office hours

- A lab instructor associated with the course
- The tutors for the department
- A tutor I pay personally
- Another class member
- A family member or roommate

7. **When a question is asked in class**
    - I am curious to hear how others answer the question
    - I am glad I don't have to answer
    - Explain if necessary
8. **During class I am**
    - intimidated by others in the class
    - an active participant in discussions
    - Explain if necessary
9. **When a new concept is presented**
    - I usually understand it right away
    - I tend to need more time grasping the concept
    - Explain if necessary
10. **When I ask a question in class**
    - I feel that others are supportive
    - I feel like others are judging my question
    - Explain if necessary
11. **When I'm struggling with the material presented**
    - I feel like I have someone to go to for help
    - I feel like I need to work it out by myself

- Explain if necessary

12. Outside of class

- I usually spend time with people in my computer class
- I usually spend time with others outside my computer class
- I prefer to be alone
- Explain if necessary

13. When another student needs help with a problem

- I want to assist them
- I want them to learn how to do it themselves
- Explain if necessary

14. When I ask a question of my fellow classmates

- I feel like other students are eager to answer my question and help
- I feel like other students think I am wasting their time
- Explain if necessary

15. Overall students in my class

- treat each other with respect
- treat each other with contempt
- Explain if necessary

16. What courses or subjects do you wish you had taken extra classes in before taking this class (select all that apply)?

- Algebra
- Calculus
- Logic
- Computer Technology

- Science (physics, biology, etc.)
- Humanities (English, art, etc.)
- Other (please specify)

17. Generally

- students are very well prepared for the computer science course
- students are somewhat prepared for the computer science course
- students are not at all prepared for the computer science course
- Explain if necessary

18. Does this course need additional prerequisites?

- strongly agree
- somewhat agree
- neutral
- somewhat disagree
- strongly disagree

19. The current prerequisites are necessary to succeed in this class.

- strongly agree
- somewhat agree
- neutral
- somewhat disagree
- strongly disagree

20. When I think about selecting a college major

- I have my major selected
- I would consider computer science
- My major will be something other than computer science

- Explain if necessary

21. My peers in computer science

- are people I relate to
- are people I don't really have much in common with
- Explain if necessary

22. Please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)

- My instructor in computer science makes learning computer science fun and challenging
- Class time is well spent and interesting
- Most class periods are lectures with little student interaction
- Assignments are fun to do and rewarding
- The feedback I receive on assignments helps me to improve
- Exams are fair and represent my knowledge
- Others seem better prepared for this class than I am
- The instructor often assumes I have knowledge which I do not have.

- Explain if necessary

23. I consider my computer science peers

- friends
- acquaintances
- Explain if necessary

24. When telling my friends about computer science

- they generally think it's cool
- they respond negatively
- they have no idea what the major entails.
- Explain if necessary

25. When I talk about computer science with my family

- they are very supportive and think it's a good idea
- they would prefer I do something else
- Explain if necessary

26. People generally think of computer programmers as

- cool; someone they want to be
- smart, but not necessarily fun
- Explain if necessary

27. On a scale from 1 to 10 where 1 is "Not well at all" and 10 is "Extremely well" how do you feel your fellow students are doing in...

- Friendliness toward other students.
- Helpfulness
- Responsiveness when asked a question
- Generally treating fellow students well

28. Please say how much you agree with the following statements (Strongly Agree; Agree; Neutral; Disagree; Strongly Disagree)

- I hope that I can find a career that does NOT require the use of computer science concepts.
- I would voluntarily take additional computer science courses if I were given the opportunity.

- I am comfortable with learning computing concepts.
- I doubt that I can solve problems by using computer applications.
- I do NOT use computing skills in my daily life.
- The challenge of solving problems using computer science appeals to me.
- Developing computing skills will be important to my career goals.
- I have little self-confidence when it comes to computing courses.
- I can make the computer do what I want it to do.
- Working on a computer is tedious and boring.
- Computers help me to organize my work better.

29. Please say how much you agree with the following statements (Strongly Agree; Agree; Neutral; Disagree; Strongly Disagree)

- I expect that learning to use computing skills will help me achieve my career goals.
- On the job, computer scientists spend a lot of time working alone.
- Women can excel in careers that involve computing.
- A student who performs well in computer science is likely to have a life outside of computers.
- Students who are skilled at computer science are less popular than other students.
- The women in my computer classes generally are NOT the top students.
- There are many computer science jobs available for qualified worke

- Knowledge of computing skills will NOT increase my chances of finding a good job.

30. What have you learned from this class that is useful?

31. What would you change about your computer science experience?

APPENDIX C: First Semester Survey 3

1. **What is your email address?**
2. **What is your student number?**
3. **What class are you attending?**
    - Colorado State University CS (Whitley)
    - Colorado State University CS (Wakefield)
    - Colorado State University CS (Wilcox)
    - Utah State University CS1 (Watson)
    - Utah State University CS1 (Cooley)
    - Brigham Young University - Idaho (CS)
    - Brigham Young University - Idaho (CS)
    - Southern Utah University (CSIS 0)
4. **Thinking about your life outside coursework (example: Family, Work, etc.), how would you rate your stress level in these areas? Would you say your life is...**
    - Not stressful at all
    - Slightly of stressful
    - Stressful
    - Really Stressful
5. **Please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral;somewhat disagree; strongly disagree)**
    - I understand the material presented.

- I feel like I can contribute to the class
- I feel supported when I have questions
- Help is there when I need it
- I feel like I have friends in the class

6. **At this point in the semester, I think I am**
    - doing well in the class
    - getting by in the class
    - not doing well at all
    - Explain if necessary
7. **My frustrations in class have the following causes (select all that apply)**
    - Assignments are confusing. I often have no idea how to begin.
    - Assignments are too time consuming
    - Assignments are not graded fairly
    - Assignments are tedious
    - The instructor assumes I have previous knowledge I don't have.
    - Exams aren't representative of my skills
    - My grades do not reflect the time I spend on the class
    - Class is boring
    - The textbook is difficult to read
8. **I enjoy this class for the following reasons (select all that apply)**
    - The instructor makes class interesting
    - I love to spend time on the assignments
    - What I am learning will be helpful on the job
    - I feel confident in my skills

- I enjoy technology.

9. When I'm struggling with the material presented I go to (Never; Once or Twice; Frequently; On an almost daily basis)

- My instructor, during office hours
- A lab instructor associated with the course
- The tutors for the department
- A tutor I pay personally
- Another class member
- A family member or roommate
- Explain if necessary

10. When another student needs help with a problem

- I want to assist them
- I want them to learn how to do it themselves
- I am unaware of others who need help.
- Explain if necessary

11. Overall students in my class

- treat each other with respect
- are not interested in others in the class
- treat each other with contempt
- Explain if necessary

12. I feel I was

- prepared for this course
- would have liked better preparation for this course
- Explain if necessary

13. My confidence in my skills in computer science is

- Very strong
- Somewhat strong
- Neutral
- Poor
- Very poor

14. What grade are you earning in this class?

15. At this point in the semester, what courses or subjects do you wish you had taken extra classes in before taking this class?

- Algebra
- Calculus
- Logic
- Science (physics, biology, etc.)
- Humanities (English, art, etc.)
- Other (please specify)

16. I generally feel

- the instructor realizes and appreciates the amount of time I spend on homework
- the instructor does not realize how much time it takes me to complete assignments
- Explain if necessary

17. The homework assigned is

- Interesting
- Tedious and uninteresting

18. **At this point in the semester, when I think about finishing a college degree**
    - It is certainly a reachable goal
    - It would take too long to be realistic for me
    - Explain if necessary
19. **This course is**
    - worth my time
    - somewhat useful
    - a waste of time
    - Explain if necessary
20. **Outside of class, how many hours per week do you spend on this class?**
    - Please Explain
21. **What have you learned from this class that is useful?**
22. **What has been your favorite computer science assignment? Why?**
23. **What would you change about your computer science experience?**
24. **Would you recommend this class to others?**
25. **Would you recommend this class to others?**
    - Yes definitely
    - Only if it was required for them
    - Perhaps
    - Definitely not
    - Please explain.

APPENDIX D: First Semester Survey 4

1. **What is your email address?**
2. **What is your student number?**
3. **What class are you attending?**
    - Colorado State University CS(Whitley)
    - Colorado State University CS(Wakefield)
    - Colorado State University CS(Wilcox)
    - Utah State University CS1 (Watson)
    - Utah State University CS1 (Cooley)
    - Brigham Young University - Idaho (CS 165)
    - Brigham Young University - Idaho (CS 124)
    - Southern Utah University (CSIS 1400)
4. **Active learners tend to retain and understand information best by doing something active with it—discussing or applying it or explaining it to others. Reflective learners prefer to think about it quietly first. How would you characterize yourself?**
    - I am clearly an Active learner
    - I am somewhat Active
    - I am neutral
    - I am somewhat Reflective
    - I am clearly a Reflective learner
5. **Sensing learners tend to be patient with details and good at memorizing facts and doing hands-on (laboratory) work; intuitive learners may be better at grasping new concepts and are often more comfortable than**

Sensing learners with abstractions and mathematical formulations. How would you identify yourself?

- I am clearly a Sensing learner
- I am somewhat Sensing
- Neutral (I do about the same with both)
- I am somewhat Intuitive
- I am clearly an intuitive learner

6. **Visual learners remember best what they see—pictures, diagrams, flow charts, time lines, films, and demonstrations. Verbal learners get more out of words—written and spoken explanations. Everyone learns more when information is presented both visually and verbally. How would you characterize yourself?**
    - I am clearly a visual learner
    - I am somewhat visual
    - Neutral (I do about the same of both)
    - I am somewhat verbal
    - I am clearly a verbal learner
7. **Sequential learners tend to gain understanding in linear steps, with each step following logically from the previous one. Global learners tend to learn in large jumps, absorbing material almost randomly without seeing connections, and then suddenly "getting it." How would you characterize yourself?**
    - I am clearly a sequential learner
    - I am somewhat sequential

- Neutral (I do about the same of both)
- I am somewhat global
- I am clearly a global learner

8. At this point in the semester, I think I am

- doing fantastic in the class
- doing better than I expected in the class
- getting by in the class
- doing worse than I do in other classes
- not doing well at all
- Explain if necessary

9. What grade are you earning in this class?

10. At this point in the semester, my confidence is

- growing steadily
- growing slowly
- declining
- Explain if necessary

11. Thinking about the course material, I generally

- understand the material deeply; I could talk about it for hours
- have a solid understanding
- understand it well enough to answer the test questions, then forget about it later

12. My level of interest in computer science has

- increased during this class
- stayed about the same

- lessened somewhat
- diminished greatly
- Explain if necessary

13. **On a scale of 1 to 5 where 1 is "Not very much work" and 5 is "Too much work" how would you rate the work load in the class?**
14. **Outside of class, how many hours per week do you spend on this class?**
15. **How many hours per week do you work at a job?**
    - I don't have a job
    - 1-10
    - 11-20
    - 21-30
    - 31-40
    - more than 40
16. **Thinking about your life outside coursework (example: Family, Work, etc.), how would you rate your stress level in these areas? Would you say your life is...**
    - Not stressful at all
    - Slightly of stressful
    - Stressful
    - Really Stressful
17. **Please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**
    - I understand the material presented.

- I feel frustrated at the level of detail required in the homework
- Homework is challenging, but rewarding.
- The homework is fairly easy for me to do
- I am spending way too much time on the homework

18. Which of the following would you suggest before taking this course?

- Boolean Logic experience
- Understanding of bits, bytes.
- Practice with algorithms
- Basic class in using computers
- Programming in a simpler language
- Web page design
- Nothing, the class needs no additional prerequisites.
- Explain if necessary

19. I feel that for the number of credits awarded for this course

- there is too much work involved
- the work involved is just about right
- Explain if necessary

20. What grade are you earning in this class?

21. Compared to other courses I am taking,

- this class takes more than twice as long as any other course
- there is too much work involved in this course
- the work involved is just about right in this course
- Explain if necessary

22. Please rate your feelings for each statement below in the following way:

(strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)

- My instructor in computer science makes learning computer science fun and challenging
- Class time is well spent and interesting
- Most class periods are lectures with little student interaction
- Assignments are fun to do and rewarding
- The feedback I receive on assignments helps me to improve
- Exams are fair and represent my knowledge

23. **Which options best describe your feelings about this course (select all that apply)?**
    - This is one of the best classes I have taken.
    - The class was challenging, but was well worth the effort.
    - I would recommend this class to others.
    - This course is a poor match for my interests.
24. **What factors influence your selection of a major (choose all that apply)?**
    - Leads to a high paying job
    - Parental desires
    - Major allows time for work schedule and social life
    - Could not get first choice major
    - Leads to a job which allows me to work from home
    - Leads to a job which has flexible hours
    - Job availability
    - Allows time for other activities

- Leads to a fulfilling career
- Leads to a job with good working conditions
- Respect for those in the field
- Have a relative in the same field
- Have friends who talked to me about the major
- Explain if necessary

25. When thinking about a college degree

- I plan to finish no matter what
- I'm not sure whether it will be something I will actually finish
- I'll see where this class takes me and go from there
- Explain if necessary

26. When I think about doing the homework assignments

- I usually wait till the last minute to get them done
- I can't wait to start them because they are so interesting
- they are too challenging and daunting...I don't know where to start
- Explain if necessary

27. It is assumed that you have learned the skills of programming. Other than programming, what have you learned from this class that is useful?

28. Have you ever shown one of your working programs to a friend or family member?

- Yes, many times.
- Yes, once or twice.
- No.
- If so, list one assignment you wanted others to see.

29. **What has been your favorite computer science assignment? Why?**
30. **What would you change about your computer science instructor?**
31. **What would you change about your computer science experience?**
32. **How likely are you to recommend this class to others?**
    - Extremely likely
    - Very likely
    - Moderately likely
    - Slightly likely
    - Not at all likely
33. **How would you finish this statement? I would be more likely to major in computer science if...**
34. **This course was**
    - worth my time
    - somewhat useful
    - a waste of time
    - Explain if necessary
35. **The following have been suggested as possible improvements to computer science program. Please rate your feelings for each statement below in the following way: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree.)**
    - It would help if there was better advising to assess my skills and interests.
    - I would learn more from group assignments or paired programming.

- Improving the quality of instruction is necessary.
- Improving the quality of tutoring and the availability of outside help is necessary.
- Classes need to use more variety in teaching methods
- Too much math is required
- There are too many non-faculty (teaching assistants, adjuncts) teaching core classes
- Many CS topics I want to study are not taught

36. What one thing would make the most difference in this course?

- A different teacher
- A different book
- Better assignments
- Better tutoring
- A slower pace
- This course needs no improvement

37. What is your major?

38. When did you decide on your current major?

- Before high school
- When I was in high school
- When I entered college
- In my sophomore year
- In my junior year
- In my senior year
- After first completing a degree in something else

39. **Because of this class, are you considering majoring in computer science?**
    - Yes. I am currently a CS major
    - Yes. I am not currently a CS major, but I am considering majoring in computer science
    - No. I have a major and I am committed to it
    - Absolutely not. I have no interest
40. **On a scale of poor to excellent, please rate your overall... (Fair; Good; Very Good; Excellent; No opinion)**
    - educational experience at your current institution
    - impression of the CS department at your current institution

APPENDIX E: Non-USU Senior Survey

1. **Clicking on the "agree" button below indicates that: you have read the above information; you voluntarily agree to participate; you are at least 18 years of age. If you do not wish to participate in the research study, please decline participation by clicking on the "disagree" button.**
    - Agree
    - Disagree
2. **What is your student number?**
3. **What is your email address (this will be used to send you future survey links and to contact you when you have completed the surveys to see where you would like the incentive mailed)?**
4. **What university do you currently attend?**
    - Utah State University
    - Southern Utah University
    - Brigham Young University Idaho
    - Colorado State University
    - Other (please specify)
5. **Are you male or female?**
    - Male
    - Female
6. **How old are you?**
7. **Where are you currently living?**
8. **When will you graduate?**
    - Within a year

- Within two years
- Within three years
- Not sure

9. **What is your major?**
10. **When did you decide on your most recent major?**
    - Before high school
    - When I was in high school
    - When I entered college
    - In my sophomore year
    - In my junior year
    - In my senior year
    - After first completing a degree in something else
    - Other comments
11. **List all the majors you had in college before you finally settled on your current major.**
12. **Knowing what you now know, would you pick the same major if you had it to do over again? Explain.**
13. **What factors influenced your choice of major (choose all that apply)?**
    - Leads to a high paying job
    - Parental desires
    - Major allows time for work schedule and social life
    - Could not get first choice major
    - Leads to a job which allows me to work from home
    - Leads to a job which has flexible hours

- Job availability
- Allows time for other activities
- Leads to a fulfilling career
- Leads to a job with good working conditions
- Respect for those in the field
- Have a relative in the same field
- Have friends who talked to me about the major
- Other (please specify)

14. **Do you currently have a job? If so, what is the job title for your current position?**
15. **In talking to others in the program, how well are they prepared for the job market?**
    - Extremely well
    - Very well
    - Moderately well
    - Slightly well
    - Not at all well
    - I have no idea
    - Please explain your ranking.
16. **Thinking about course material in my major, I generally**
    - understand the material deeply; I could talk about it for hours
    - have a moderate understanding
    - understand it well enough to answer the test questions, then forget about it later

17. **Thinking about courses in your major and how they were taught, generally**
    - they were more lecture; few if any activities or demonstrations
    - quite a few activities and demonstrations; less lecturing
18. **Compared to those at other institutions, how effective is the education you received?**
    - Much more effective
    - Somewhat more effective
    - Slightly more effective
    - About as effective
    - Slightly less effective
    - Somewhat less effective
    - Much less effective
    - I have no idea
19. **What suggestions do you have for improving the education provided?**
20. **How likely are you to recommend your major to others?**
    - Extremely likely
    - Very likely
    - Moderately likely
    - Slightly likely
    - Not at all likely
    - Please explain.
21. **How likely are you to recommend your college or university to others?**
    - Extremely likely

- Very likely
- Moderately likely
- Slightly likely
- Not at all likely
- Please explain.

22. **Are you aware of individuals who were in your same major but have changed majors?**
    - Yes, that is very common
    - Yes, but it is not common
    - No
    - In your opinion, why do people switch from your current major?
23. **If you wanted to convince someone to pick your same field as a profession, what would you say to convince them?**
24. **What are the least desirable things about your major?**
25. **In your opinion, why don't more students pick the major you chose in college?**
26. **What is your GPA (on a 4 point scale)?**
    - 3.6 - 4.0
    - 3.1 - 3.5
    - 2.6 - 3.0
    - 2.1 - 2.5
    - 2.0 or below
    - Other (please specify)
27. **Have you taken a computer science class?**

- Yes, it was my major.
- Yes, it was my minor.
- Yes, I took several as they were required for my major
- Yes, I took one.
- No

28. **The following have been suggested as possible improvements to the computer science program. Please rate your feelings for each statement below regarding your computer science classes. Please use the following rating: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**
    - My instructors in computer science make learning computer science fun and challenging
    - Class time is well spent and interesting
    - Most class periods are lectures with little student interaction
    - Assignments are fun to do and rewarding
    - The feedback I receive on assignments helps me to improve
    - Exams are fair and represent my knowledge
29. **We are looking for ways to improve the computer science program. Please rate your feelings for each statement below regarding your computer science classes. Please use the following rating: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**
    - It would help if there was better advising to assess my skills and interests.
    - I would learn more from group assignments or paired programming.

- Improving the quality of instruction is necessary.
- Improving the quality of tutoring and the availability of outside help is necessary.
- Classes need to use more variety in teaching methods

APPENDIX F: USU Senior Survey

1. **Clicking on the "agree" button below indicates that: you have read the above information; you voluntarily agree to participate; you are at least 18 years of age. If you do not wish to participate in the research study, please decline participation by clicking on the "disagree" button.**
    - Agree
    - Disagree
2. **What is your student number?**
3. **What is your email address**
4. **Are you male or female?**
    - Male
    - Female
5. **What is your class rank?**
    - Freshman
    - Sophomore
    - Junior
    - Senior
    - Other
6. **What is your GPA (on a 4 point scale)?**
    - 3.6 - 4.0
    - 3.1 - 3.5
    - 2.6 - 3.0
    - 2.1 - 2.5
    - 2.0 or below

7. **What is your major?**
8. **When did you decide on your most recent major?**
    - Before high school
    - When I was in high school
    - When I entered college
    - In my sophomore year
    - In my junior year
    - In my senior year
    - After first completing a degree in something else
    - Other comments
9. **List all the majors you had in college before you finally settled on your current major.**
10. **Knowing what you now know, would you pick the same major if you had it to do over again? Explain.**
11. **What factors influenced your choice of major (choose all that apply)?**
    - Leads to a high paying job
    - Parental desires
    - Major allows time for work schedule and social life
    - Could not get first choice major
    - Leads to a job which allows me to work from home
    - Leads to a job which has flexible hours
    - Job availability
    - Allows time for other activities
    - Leads to a fulfilling career

- Leads to a job with good working conditions
- Respect for those in the field
- Have a relative in the same field
- Have friends who talked to me about the major

12. **How old are you?**
13. **Do you currently have a job? If so, what is the job title for your current position?**
14. **In talking to others in the program, how well are they prepared for the job market?**
    - Extremely well
    - Very well
    - Moderately well
    - Slightly well
    - Not at all well
    - I have no idea
    - Please explain your ranking.
15. **Thinking about course material in my major, I generally**
    - understand the material deeply; I could talk about it for hours
    - have a moderate understanding
    - understand it well enough to answer the test questions, then forget about it later
16. **Thinking about courses in your major and how they were taught, generally**
    - they were more lecture; few if any activities or demonstrations

- quite a few activities and demonstrations; less lecturing

17. **Compared to those at other institutions, how effective is the education you received?**
    - Much more effective
    - Somewhat more effective
    - Slightly more effective
    - About as effective
    - Slightly less effective
    - Somewhat less effective
    - Much less effective
    - I have no idea
18. **What suggestions do you have for improving the education provided?**
19. **How likely are you to recommend your major to others?**
    - Extremely likely
    - Very likely
    - Moderately likely
    - Slightly likely
    - Not at all likely
    - Please explain.
20. **Are you aware of individuals who were in your same major but have changed majors?**
    - Yes, that is very common
    - Yes, but it is not common
    - No

- In your opinion, why do people switch from your current major?

21. **If you wanted to convince someone to pick your same field as a profession, what would you say to convince them?**
22. **What are the least desirable things about your major?**
23. **In your opinion, what would be the best way to increase the number of students in your major?**
24. **In your opinion, why don't more students pick the major you chose in college?**
25. **The following have been suggested as possible improvements to the computer science program. Please rate your feelings for each statement below regarding your computer science classes. Please use the following rating: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**
    - My instructors in computer science make learning computer science fun and challenging
    - Class time is well spent and interesting
    - Most class periods are lectures with little student interaction
    - Assignments are fun to do and rewarding
    - The feedback I receive on assignments helps me to improve
    - Exams are fair and represent my knowledge
26. **We are looking for ways to improve the computer science program. Please rate your feelings for each statement below regarding your computer science classes. Please use the following rating: (strongly agree; somewhat agree; neutral; somewhat disagree; strongly disagree)**

- It would help if there was better advising to assess my skills and interests.
- I would learn more from group assignments or paired programming.

- Improving the quality of instruction is necessary.
- Improving the quality of tutoring and the availability of outside help is necessary.
- Classes need to use more variety in teaching methods

APPENDIX G: Professional Survey

1. **Clicking on the "agree" button below indicates that: you have read the above information; you voluntarily agree to participate; you are at least 18 years of age. If you do not wish to participate in the research study, please decline participation by clicking on the "disagree" button.**
    - Agree
    - Disagree
2. **What is your name?**
3. **Gender**
    - Male
    - Female
4. **What is your preferred email address?**
5. **In what state or U.S. territory are you currently living?**
6. **What is your birth year?**
7. **List all universities you have attended (in chronological order).**
8. **What is the highest degree you have completed?**
    - Some college but no degree
    - Associate degree
    - Bachelor degree
    - Masters degree
    - PhD
    - Post Graduate Work
9. **List all the majors you had in college (ending with the major of your most recent degree)**

10. **When did you decide on the discipline of your final degree?**
    - Before high school
    - When I was in high school
    - When I entered college
    - In my sophomore year
    - In my junior year
    - In my senior year
    - After first completing a degree in something else
    - After entering the job market
11. **What company do you currently work for?**
12. **List all the companies you have previously worked for (since graduation) in chronological order.**
13. **What is the job title for your current position?**
14. **Which of the following best describes your current position? You may list up to Please list them in order with the closest match first. If you cannot find items in the list that match your profession, choose "other" and fill in the description in the next question.**
15. **If you chose "Other" in the previous question, please input your response in the corresponding textbox below:**
16. **Which of the following categories best describes your employment status?**
    - Employed, working 1-20 hours per week
    - Employed, working 21-39 hours per week
    - Employed, working 40 or more hours per week

- Not employed, looking for work
- Not employed, NOT looking for work
- Retired
- Disabled, not able to work

17. About how long have you been in your current position?

18. About how many employees work at your company?

19. How well did your education prepare you for your current job?

- Extremely well
- Very well
- Moderately well
- Slightly well
- Not at all well
- Please explain your ranking.

20. Compared to other employees' education, how effective is the education you received?

- Much more effective
- Somewhat more effective
- Slightly more effective
- About as effective
- Slightly less effective
- Somewhat less effective
- Much less effective
- Please explain.

21. How likely are you to recommend your field of work to others?

- Extremely likely
- Very likely
- Moderately likely
- Slightly likely
- Not at all likely

22. How often do you talk to young people about your work?

- Almost never
- Occasionally
- Often
- I regularly engage in such outreach activities

23. Have you ever seriously considered employment in a non-computer science field? Explain.

24. Are you aware of individuals with your same training who have taken employment in an unrelated area?

- Yes, that is very common
- Yes, but it is not common
- No
- Why do you think people have sought other employment?

25. What are the best things about your job?

26. What are the least desirable things about your job?

27. In your opinion, what would be the best way to increase the number of students in your field of expertise?

28. In your opinion, why don't more students pick your field of expertise?

29. What is the last year you attended USU?

30. **If you completed an undergraduate degree at USU, which specialization, track or area of emphasis did you complete?**
    - Bioinformatics
    - Digital Systems
    - Information Technology
    - Science
    - Software Development
    - Don’t remember
    - Other (please specify)
31. **Which of the following statements best describe your educational pursuits one year after earning your BS degree?**
    - I completed an advanced degree.
    - I pursued an advanced degree as a full-time student.
    - I pursued an advanced degree as a part-time student.
    - I continued to take a few classes as an un-matriculated student.
    - I did not attend any school.
    - N/A – It’s has not been a year yet since graduation
32. **Which of the following best describes your employment situation one year AFTER graduation from the BS program?**
    - Full time (hours/week) in a CS-related position
    - Full time (hours/week) in a non-CS-related position
    - Part-time (anything less than 35 hours/week) in a CS-related position
    - Part-time (anything less than 35 hours/week) in a non-CS-related position

- Unemployed
- N/A – it's hasn't been a year since graduation

33. **What was your starting pay rate for your first job after earning your BS degree?**
    - Amount per year
    - Amount per month
    - Amount per hour
    - Decline to say
    - Nothing; not employed immediately following graduation
    - Please specify amount and year (Ex: $60,in 2009)
34. **Which CS classes from your BS degree best helped you prepare for your professional career or subsequent educational pursuits? You may list up to Please list them in order, with most helpful first.**
35. **Which CS professors were the most influential in your education and career? (select at most four)**
    - Vicki Allan
    - Stephen Allan
    - Daniel Bryce
    - Renee Bryce
    - Scott Cannon
    - Heng-Da Cheng
    - Stephen Clyde
    - Donald Cooley
    - Nelson Dinnerstein

- Kendra Dinnerstein
- Linda DuHadway
- Curtis Dyreson
- Del Dyerson
- Larry Egbert
- Robert Erbacher
- Nicholas Flann
- Rex Hurst
- Minghui Jiang
- Greg Jones
- Mary Veronika Kolesar
- Vladimir Kulyukin
- Ming Li
- SeungJin Lim
- Chad Mano
- Dean Mathias
- Neil Morgan
- Wendel Pope
- Xiaojun Qi
- Kenneth Sunberg
- Haitao Wang
- Daniel Watson
- Others
- Please Explain

36. **Which topics were not adequately covered in your degree program? You may list zero or more. Please list them in order, with least adequately covered topic first.**
37. **Which topics were OVER emphasized in your degree program? You may list zero or more.**
38. **On a scale of poor to excellent, please rate your overall... (Poor Fair Good Very Good Excellent No opinion)**
    - educational experience at USU.
    - impression of the USU/CS department at the time of your most recent graduation.
    - impression of the current USU CS department and its degree programs.
39. **Would you recommend the BS degree program in Computer Science at USU to students interested in computer science?**
    - Yes
    - No
    - No opinion
40. **Please share with us your recommendations about the curriculum?**
41. **Do you have any other comments about the CS department?**

APPENDIX H: Tutor Survey

1. **If you would like compensation, please provide your name.**
2. **Clicking on the "agree" button below indicates that: you have read the above information; you voluntarily agree to participate; you are at least 18 years of age. If you do not wish to participate in the research study, please decline participation by clicking on the "disagree" button.**
    - Agree
    - Disagree
3. **Are the required math classes utilized in computer science classes?**
    - Strongly Agree
    - Agree
    - Neutral
    - Disagree
    - Strongly Disagree
    - Why or why not?
4. **It has been said that math helps the student think logically. What other classes could be used to aid logical thinking?**
5. **Some students say that the workload in computer science is too much. From what you see in the tutor room, is that generally true? To what degree is poor preparation or poor skills a contributing factor?**
    - Yes
    - No

- Why or why not?

6. **What would you do to decrease the workload in computer science while still teaching the same principles?**
7. **Based on your experience, at what point in the FIRST semester are students the most likely to feel the most stress from their schoolwork?**
    - First month
    - Second month
    - Third month
    - Toward the end of the semester
    - Other (please specify)
8. **Based on your experience, at what point in the ANY semester are students the most likely to feel the most stress from their schoolwork?**
    - First month
    - Second month
    - Third month
    - Toward the end of the semester
    - Why or why not?
9. **To what extent is the teacher responsible for learning difficulties? Do you see significant differences between classes? Are the differences a result of teacher skills or merely assignment difficulty?**
10. **Suppose you had a friend wanting to take CSWho would you recommend they take the course from (assuming the person was teaching the class)? You can select more than one.**

- Vicki Allan
- Scott Cannon
- Heng-da Cheng
- Stephen Clyde
- Donald Cooley
- Linda DuHadway
- Curtis Dyerson
- Nick Flann
- Minghui Jiang
- Vladimir Kulyukin
- Xiaojun Qi
- Haitao Wang
- Ming Li
- Dan Watson
- Dean Mathias
- Ken Sundberg
- Why?

11. **Some students indicate that they have absolutely no idea how to begin an assignment. Do you see that? if so, what would help?**
    - Yes
    - No
    - What would help?
12. **What specific complaints do you hear from students regarding their instructors?**

13. **What specific praise do you hear from students regarding their instructors?**
14. **In a previous study, some students indicated that computer science assignments could be improved. Based on your experience assisting students with their assignments, what general changes would you make to the assignments?**
15. **In a previous study, some students thought it would be helpful to start students in a simpler language than C++, C or Java. Do you agree?**
    - Yes
    - No
    - Why or why not?
16. **What language(s) would you choose for beginning students?**
    - Visual Basic
    - JavaScript
    - PHP
    - Ruby
    - Python
    - Pascal
    - PERL
    - Java
    - C#
    - C++
    - C
    - Other (please specify)

17. **Of the languages you chose in the previous question, which language do you think would be the BEST beginning language?**
    - Visual Basic
    - JavaScript
    - PHP
    - Ruby
    - Python
    - Pascal
    - PERL
    - Java
    - C#
    - C++
    - C
    - Other (please specify)
18. **It has been stated in many studies that computer science students have a social stigma attached (i.e. geek, nerd, etc.). What would you do to help potential students understand what computer science really is?**
19. **In a previous survey, students have indicated that when they need help, there is little available. What changes would you make to help students when they need help?**
20. **Many students say they never go to the tutor room. What pattern of tutor room use do you observe?**

21. **Students often complain that assignments are busy work. Give an example of an assignment that seemed to be of little value.**
22. **Students often complain that assignments are busy work. Give an example of an assignment that seemed to be of GREAT value.**
23. **What characteristics of students seem to be the best predictor of their success in computer science?**
24. **What would you change about the computer science department? (i.e. what is the department doing poorly?)**
25. **Thinking of the computer science department overall, what would you the department is doing well?**
26. **Is there anything else you would like to mention about how the computer science department could improve?**

APPENDIX I: Women in Computing

1. **Application ID**
2. **What year in school are you?**
    - Senior
    - Junior
    - Sophomore
    - Freshman
3. **Which of the following computing activities/concepts have you learned, or participated in, either through school, in an after-school/summer program, or on your own? Tell us how involved you have been by using the rating scale: Not at all, A little, Pretty much, A lot.**
    - Word Processing (e.g., MS Word, WordPerfect, Open Office)
    - Spreadsheets (e.g., MS Excel, Open Office, IWork)
    - Presentations (e.g., MS Powerpoint, Keynote, Prezi, Open Office)
    - Desktop Publishing (e.g., MS Publisher, InDesign, Pagemaker)
    - Mathematical Applications (e.g., MatLab, Mathcad, Mathematica)
    - Web Design/Development (e.g., Dreamweaver, MS Expression, Web Studio)
    - Multimedia Design (e.g., Photoshop, Imovie, Premiere, Movie Maker, Illustrator)
    - Computer Aided Design (e.g., Autocad, Inventor 3D)
    - Game design (GameMaker, Blender, Maya, Visual Studio/Basic/C++, C#)
    - Mobile application development (Objective C, AppInventor,

Iphone/Android, Ipad/Tablet)

- Web application development (Facebook Api, Flash, Java, HTML5, PHP)
- Computer Technology (e.g., tech crew, install/maintain computers for yourself and others, software installation)
- Network/System Operations, Maintenance (e.g., system manager for multiple machines)
- Network design/development (e.g., LAN, wireless)
- Network security
- Robotics (e.g., Labview, ROBOTC, LEGO Mindstorms)
- Graphic (e.g., Scratch, Alice, Flash, OpenGL, Java 3d)
- In a coding language (e.g., Visual Basic, Python, Java, C++)
- Using variables, loops, decision logic, arrays/lists
- Using advanced data structures (stacks, queues, trees)
- Mainframe programming
- Database design/programming (Access, MySQL, Filemaker)
- Open source contributions (contributing original code segments to open source libraries)

4. **In which of the following extra-curricular (out-of-school) computing and information technology activities have you participated? Tell us how involved you have been by using the rating scale: Not at all, A little, Pretty much, A lot.**

    - Computing/Technology-related club or team (at school or outside of school)

- Computing/Technology-related community involvement/volunteerism

- College-level computing/information technology courses
- Independent study of computing/technology to advance your knowledge

- Computing/Technology-related job
- Computing/Technology-related summer or after-school program
- Computing/Technology-related internship

5. **What is your GPA?**
6. **What are your career plans or area of interest for after high school?**
    - Computer Graphics
    - Computer Programming
    - Computer Science
    - Computer Software Engineering
    - Business
    - Computer Networking and Telecommunications
    - Artificial Intelligence and Robotics
    - Biological and Biomedical Sciences
    - Science Technologies
    - Multi/Interdisciplinary Studies
    - Web Development
    - Computer Forensics
    - Computer and Information Security
    - Game Design and Development

- Parks, Recreation, and Fitness
- Languages, Literatures, and Linguistics
- Arts, Visual and Performing
- Health Professions and Related Clinical Sciences
- Engineering (other)
- Undecided

7. **What have you done with computing and technology? Please also describe the computing-related accomplishment that makes you most proud.**
8. **Please describe ways in which you show leadership in computing, leadership in your school, and/or leadership in your community.**
9. **How do you see your aspirations in computing and technology influencing your future pursuit.**
10. **Underrepresented status (optional, self-declared by each applicant)**
    - American Indian or Alaskan Native
    - Asian / Pacific Islander
    - Black or African American
    - Hispanic / Latina
    - White / Caucasian
11. **Socio-economic status (determined by % subsidized lunch at school)**
12. **Notes of support from a school official**

APPENDIX J: Learning Styles Survey (Asked of all Groups)

1. **I find an activity more enjoyable if there is competition.**
    - Strongly agree
    - Agree
    - Disagree
    - Strongly disagree
2. **In solving a problem, I generally need a lot of help.**
    - Strongly agree
    - Agree
    - Disagree
    - Strongly disagree
3. **In solving a problem, I prefer to work in groups.**
    - Strongly agree
    - Agree
    - Disagree
    - Strongly disagree
4. **While solving a difficult problem, what kind of help would you like?**
    - None
    - Feedback if final solution is correct or not
    - Correct solution given on completion
    - Periodic feedback about sub problems
    - Suggestions from tutor
5. **I understand something better after I**
    - try it out.
    - think it through.

6. **I would rather be considered**
    - realistic.
    - innovative.
7. **When I think about what I did yesterday, I am most likely to get**
    - a picture.
    - words.
8. **I tend to**
    - understand details of a subject but may be fuzzy about its overall structure.
    - understand the overall structure but may be fuzzy about details.
9. **When I am learning something new, it helps me to**
    - talk about it.
    - think about it.
10. **If I were a teacher, I would rather teach a course**
    - that deals with facts and real life situations.
    - that deals with ideas and theories.
11. **I prefer to get new information in**
    - pictures, diagrams, graphs, or maps.
    - written directions or verbal information.
12. **Once I understand**
    - all the parts, I understand the whole thing.
    - the whole thing, I see how the parts fit.
13. **In a study group working on difficult material, I am more likely to**
    - jump in and contribute ideas.

- sit back and listen.

14. I find it easier

- to learn facts.
- to learn concepts.

15. In a book with lots of pictures and charts, I am likely to

- look over the pictures and charts carefully.
- focus on the written text.

16. When I solve math problems

- I usually work my way to the solutions one step at a time.
- I often just see the solutions but then have to struggle to figure out the steps to get to them.

17. In classes I have taken

- I have usually gotten to know many of the students.
- I have rarely gotten to know many of the students.

18. In reading nonfiction, I prefer

- something that teaches me new facts or tells me how to do something.
- something that gives me new ideas to think about.

19. I like teachers

- who put a lot of diagrams on the board.
- who spend a lot of time explaining.

20. When I'm analyzing a story or a novel

- I think of the incidents and try to put them together to figure out the themes.
- I just know what the themes are when I finish reading and then I have to

go back and find the incidents that demonstrate them.

21. When I start a homework problem, I am more likely to

- start working on the solution immediately.
- try to fully understand the problem first.

22. I prefer the idea of

- certainty.
- theory.

23. I remember best

- what I see.
- what I hear.

24. It is more important to me that an instructor

- lay out the material in clear sequential steps.
- give me an overall picture and relate the material to other subjects.

25. I prefer to study

- in a study group.
- alone.

26. I am more likely to be considered

- outgoing.
- reserved.

27. When I get directions to a new place, I prefer

- a map.
- written instructions.

28. I learn

- at a fairly regular pace. If I study hard, I'll "get it."

- in fits and starts. I'll be totally confused and then suddenly it all "clicks."

29. I would rather first

- try things out.
- think about how I'm going to do it.

30. When I am reading for enjoyment, I like writers to

- clearly say what they mean.
- say things in creative, interesting ways.

31. When I see a diagram or sketch in class, I am most likely to remember

- the picture.
- what the instructor said about it.

32. When considering a body of information, I am more likely to

- focus on details and miss the big picture.
- try to understand the big picture before getting into the details.

33. I more easily remember

- something I have done.
- something I have thought a lot about.

34. When I have to perform a task, I prefer to

- master one way of doing it.
- come up with new ways of doing it.

35. When someone is showing me data, I prefer

- charts or graphs.
- text summarizing the results.

36. When writing a paper, I am more likely to

- work on (think about or write) the beginning of the paper and progress

forward.

- work on (think about or write) different parts of the paper and then order them.

37. When I have to work on a group project, I first want to

- have "group brainstorming" where everyone contributes ideas.
- brainstorm individually and then come together as a group to compare ideas.

38. I consider it higher praise to call someone

- sensible.
- imaginative.

39. When I meet people at a party, I am more likely to remember

- what they looked like.
- what they said about themselves.

40. When I am learning a new subject, I prefer to

- stay focused on that subject, learning as much about it as I can.
- try to make connections between that subject and related subjects.

41. I prefer courses that emphasize

- concrete material (facts, data).
- abstract material (concepts, theories).

42. For entertainment, I would rather

- watch television.
- read a book.

43. Some teachers start their lectures with an outline of what they will cover. Such outlines are

- somewhat helpful to me.
- very helpful to me.

44. The idea of doing homework in groups, with one grade for the entire group,

- appeals to me.
- does not appeal to me.

45. When I am doing long calculations,

- I tend to repeat all my steps and check my work carefully.
- I find checking my work tiresome and have to force myself to do it.

46. I tend to picture places I have been

- easily and fairly accurately.
- with difficulty and without much detail.

47. When solving problems in a group, I would be more likely to

- think of the steps in the solution process.
- think of possible consequences or applications of the solution in a wide range of areas.