2014

The Effects of Robots on Computer Science Perceptions

Shannon Marie Gray
Bard College

The Effects of Robots on Computer Science Perceptions

A Senior Project submitted to
The Division of Science, Mathematics and Computing
of Bard College
by
Shannon Gray

Annandale-on-Hudson, New York
May 2014

ABSTRACT

This project examines and analyses the data from several Institute for Personal Robots in Education (IPRE) studies in 2007, 2009 and 2010 that judged students attitudes of computer science before and after an introductory computer science course. The course was run in two conditions: a class involving robots and a class with no robots. The 2007 studies found that students in the robot classes did 10% better on average in the course than those students in the non-robots class. The data from the most recent studies show that while robots may increase performance in the class, they did not seem to have any significant effects on changing attitudes or perceptions. However, robots did increase the likelihood that students discussed their assignments and lectures with their peers, a factor that could lead to increasing involvement in computer science. This senior project aims to take their research further with another follow-up study that address two research questions. How much computer science can someone actually learn in an hour, and is an hour enough to change peoples' perceptions of computer science? Secondly, what are the differences between using robots to teach computer science versus using robot simulations? To answer these questions hour-long coding workshops were designed based on the principles of Hour of Code and the Institute for Personal Robots in Education, which would introduce participants to programming a Scribbler robot.

CONTENTS

ACKNOWLEDGEMENTS

The success of any project depends on the support and teachings of many others. I would like to express my gratitude to the people who have aided in the successful completion of this project.

I would like to thank Keith O'Hara, my advisor since first year and my senior project advisor as well. Thanks for keeping me from freaking out too often, and getting through this last year.

I would like to thank my family and friends for the support though the years, especially my mother. You have always been there for me, and it really means the world. So, thanks to all.

1
INTRODUCTION

1.1 Motivation

One goal of this project is to examine the differences between classes that use robots to teach introductory computer science, and classes that do not make use of robots. Does one have a greater effect on changing computer science perceptions? Is one more interesting than the other, or does one make a better teaching tool?

This senior project then aims to take their research further with another follow-up study that address two research questions. How much computer science can someone actually learn in an hour, and is an hour enough to change peoples' perceptions of computer science? Secondly, what are the differences between using robots to teach computer science versus using robot simulations? Does the physicality of real robots promote learning that is more engaging? Do simulations make a task more interesting or challenging? What are the differences in these two teaching tools, and how do they effect perceptions of computer science?

A broader goal of this project is to show people that anyone can—and should—do computer science. It is not as scary as it seems. You do not have to be a technology or math person. Part of the motivation is changing the reputation that computer science has and getting more people interested in computer science, an ongoing effort for many in the computer science field. Computer science and "coding" are not the same thing. Programming is an aspect of computer science and not everyone ends up a programmer. There are multitudes of fields that make use of computer science, and everyone could benefit from learning more about it.[12]

1.2 Computer Science Perceptions

Computer scientists are technology-oriented, with strong interests in programming and electronics, and little interest in people. They are so focused on technology that they are obsessed with computers and programming, to the exclusion of other interests. They were "born coding" or "dream in code". They lack interpersonal skills and are socially awkward. Computer scientists are all nerds or geeks. They are mostly unattractive, pale and thin, and wearing glasses. They are predominately male. These are just some of the common stereotypes that Cheryan et al. (2013) found in their research. Cheryan et al. (2013) also found that these stereotypes about computer scientists' are more prevalent among populations with less computing experience.[3]

1.2.1 Changing Perceptions

How do you go about changing the way something is, and has been perceived for many generations? How do you undo the stigmas associated with Computer Science? One way is to expose people to computer science; show people what computer science really is. Show them real people who have studied it. Who you are does not define what you can or cannot do, and vice versa. Instead of focusing on the typical pale, white male role models we all know, we

should learn about people like Marissa Mayer[1], Sundar Pichai[2], Elena Silenok[3], and Ginni Rometty[4]. They are defying stereotypes and can help inspire others to as well. This is what the Hour of Code Campaign embodies. Their promotional videos[5] are essentially about how important computer science is to know, and why everyone should try it.

Hour of Code[6] is an opportunity for every student to try computer science for one hour, although they encourage learning the Hour of Code all year-round.[24] During Computer Science Education Week (December 8-14, 2014), over thirty three million people participated in the Hour of Code.[8] They have tutorials and ideas for people of all ages to try, that cover a wide variety of activities: game and app designing, robotics, creating cards and graphics, et cetera. They even have activities for people that do not have computer access, but can still learn programming concepts "unplugged".[4]

The Hour of Code is not out to educate people on all the ins and outs of computer science. It is an effort to show people the essence of computer science. They want people to think about things in their everyday lives that use computer science: a cell phone, a microwave, a computer, a traffic light, etc. and that all of these things needed a computer scientist to help build them. Hour of Code aims to show that computer science is the art of blending human ideas and digital tools to increase our power. That computer scientists work in so many different areas: writing apps for phones, curing diseases, creating animated movies, working on social media, building robots that explore other planets and so much more. Again, not everyone who studies

---

[1] Current President and CEO of Yahoo!.

[2] Senior vice president at Google. Oversees Android, Chrome and Google Apps.

[3] Founder of Clothia, a fashion website and an iPad app that allows users to create virtual wardrobes, browse their friends' closets, mix-n-match items to create outfits, get inspired by style icons, share their finds, and virtually try on clothes using augmented reality via their webcam.

[4] Current Chairwoman and CEO of IBM, and the first woman to head the company.

[5] http://www.youtube.com/user/CodeOrg

[6] https://code.org/

computer science ends up a programmer, and they certainly are not all pale, unattractive antisocial people living in dark basements, despite what the stereotypes say.

2
INSTITUTE FOR PERSONAL ROBOTS IN EDUCATION

2.1 Overview

The Institute for Personal Robots in Education (IPRE) aims to change computer science perceptions and attitudes, and increase the number of people involved in computing. They believe that the use of personal robots and engaging examples can provide a sound foundation for learning computing, in addition to attracting a more diverse body of students into the computing disciplines.[12] IPRE was created to make computer science education more fun and effective through the use of personal robots, and is a joint effort between Georgia Tech and Bryn Mawr College sponsored by Microsoft Research.[10] Robots can provide a tangible and personal means to engage a student in engineering, science and math education. Their mission is broad: to employ robots in education at all levels from middle school to graduate school, though their initial audience was introductory undergraduate computer science. IPRE aims to improve retention in and attraction of students to computer science.

IPRE's program contains five main components: curriculum, community, assessment, hardware and software:

- A curriculum that can be quickly and easily adapted by others to suit their needs.
- An open community for sharing idea and supporting each other.
- Assessments of the impact and effectiveness of the tools that IPRE develops and deploys.
- Development and deployment of low cost robot for teachers and students.
- An easily accessible software environment for students to learn to program and for teachers to teach the curriculum.[9]

Utilizing these components, IPRE has run several studies with Robots and CS1 at various colleges using Myro and Scribbler robots.

2.2 Myro and Scribblers

My Robotics (Myro), is an open-source library and application programming interface aimed at making it easy for beginners to learn about computer science by programming robots. Myro has been implemented in several programming languages, including: Python, C++, C, Java, Ruby and Scheme. Myro is an excellent tool for controlling both real and virtual Scribbler robots.[14]

Scribblers are inexpensive differential[7] steer robots, manufactured by Parallax, that make an ideal platform for people to learn about robots and computer science. They include enough built-in sensors to perform a variety of common robot behaviors out-of-the-box including line-following, obstacle avoidance, and light seeking. Scribblers have two infrared detectors on the bottom of the robot, and two infrared distance or obstacle detector on the front left and right of the robot, that give it the ability to check the left and right sides of the robot for obstacles and

[7] A differential wheeled robot is a mobile robot whose movement is based on two separately driven wheels placed on either side of the robot body. It can thus change its direction by varying the relative rate of rotation of its wheels and hence does not require an additional steering motion. Unlike the wheels on a car, for instance, they cannot change angles.

react accordingly (see Figures 2.1 and 2.2). The robots also have three photocell detectors, or light sensors, on the front side of the robot, and a speaker for tone and melody generation.[21]

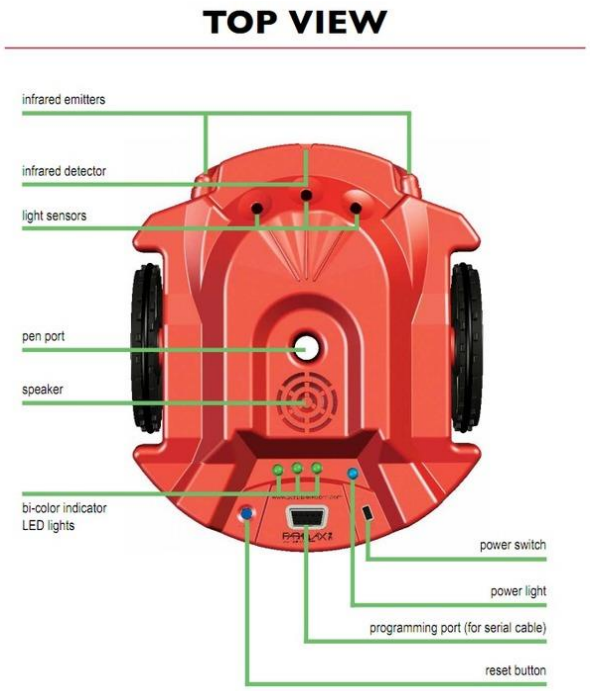


Figure 2.1. Top view of Scribbler2 robot.

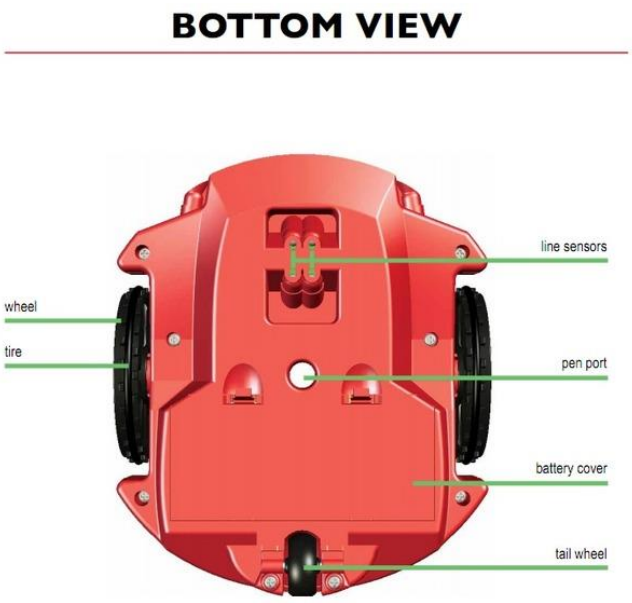


Figure 2.2. Bottom view of Scribbler2 robot.

The Scribblers are controlled with computers through a Bluetooth connection to another piece of hardware that is attached to the Scribblers, the Fluke. The Fluke and Fluke2 (like the Scribblers, there is a second version) were also designed by IPRE and are connected to the Scribblers via the serial programming port. The Flukes have a camera, three IR emitters, one IR receiver, and a Bluetooth transmitter and receiver (see Figure 2.3). It draws its power directly through the programming port.[19] Additionally, Scribblers come with a simple gamepad that can be used to control the movement of the robot. The controls on the gamepad are pre-programmed, but can be modified by users.

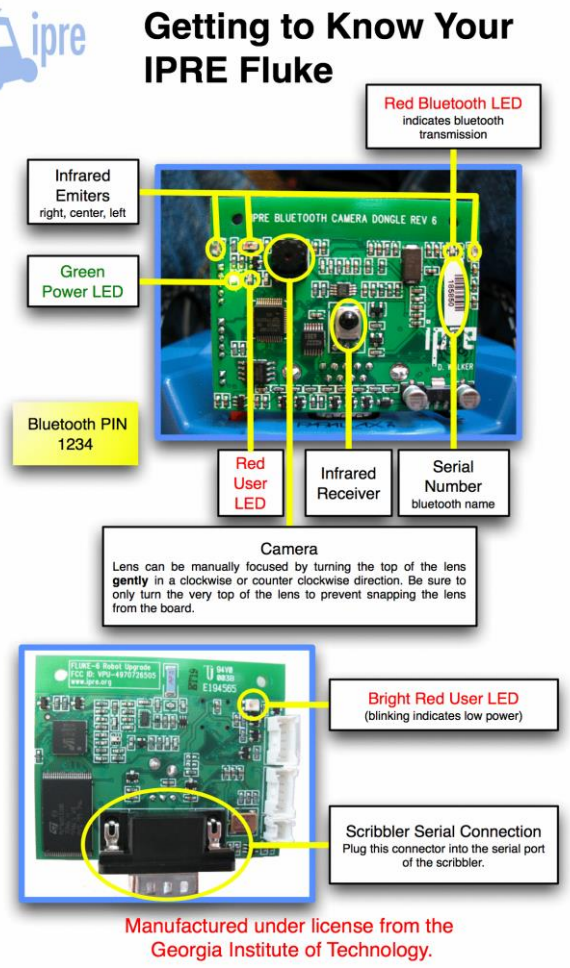


Figure 2.3. Organization of the Fluke2.

2.3 Calico

IPRE's latest interface for interacting with Myro and the Scribblers is Calico. Calico is a multi-language, multi-context programming framework and learning environment for computing education created by IPRE. Calico is completely free and open source, so anyone can download the code and use it.[8] They can even edit the code and create modules, libraries and other useful additions themselves. Calico comes with the Myro libraries pre-installed, thus there is no need

[8] http://calico.codeplex.com/

to download and install them separately. It was designed to support several different programming languages including Python, Scheme and Jigsaw, among others. Calico can be used for a variety of contexts including scientific visualization, robotics, and art. It has a shell window, output window and an integrated code editor. Calico also supports a variety of different robots, such as the Scribblers.[2]

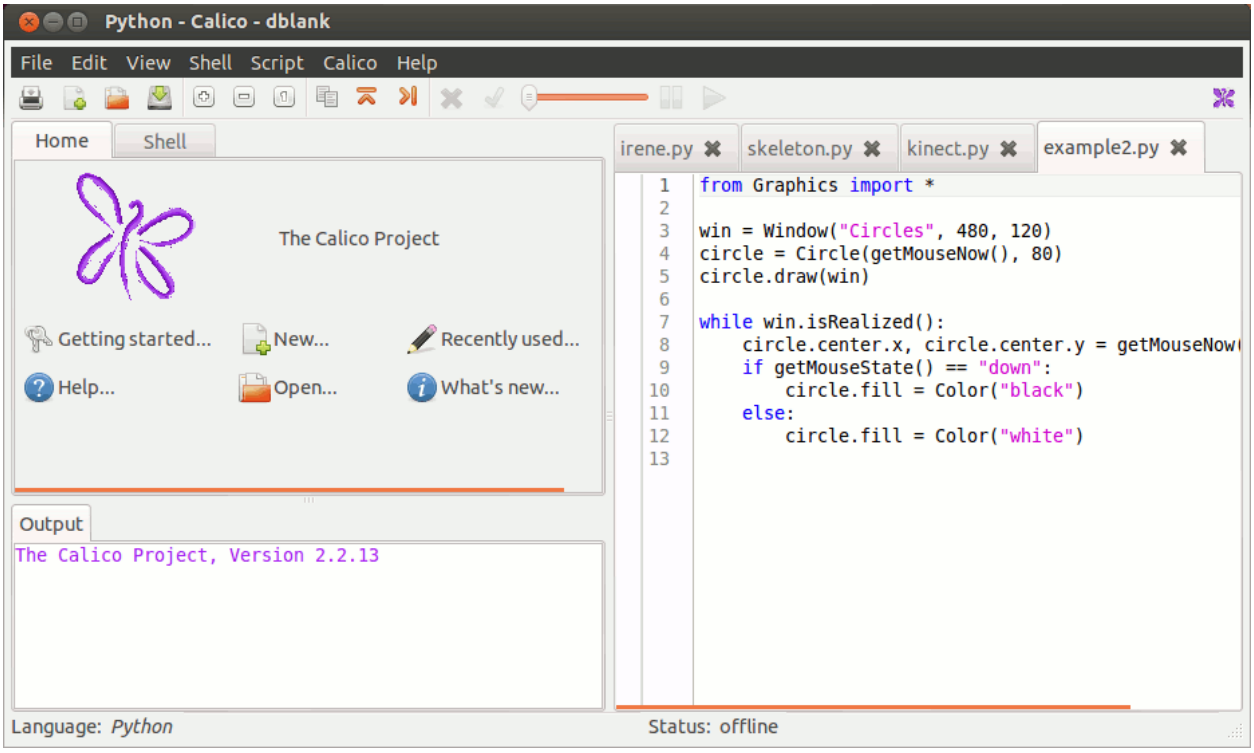


Figure 2.4. Calico Interface

Calico's Myro comes with a built in simulator (see Figure 2.5). The simulator is still under development, but allows a wide range of control over the robots and environment. The simulator allows one to control a virtual Scribbler robot, with all of the same detectors and sensors as the real robot has, including the camera. The color and arrangement of the virtual world can be changed by adding objects or lights, and controlling the starting place of the robot within the environment. The simulator can be controlled through code or the gamepad like the

real Scribblers. The simulated scribblers can accomplish most anything that the real Scribblers can, with the added functionality of being able to control multiple robots at once.

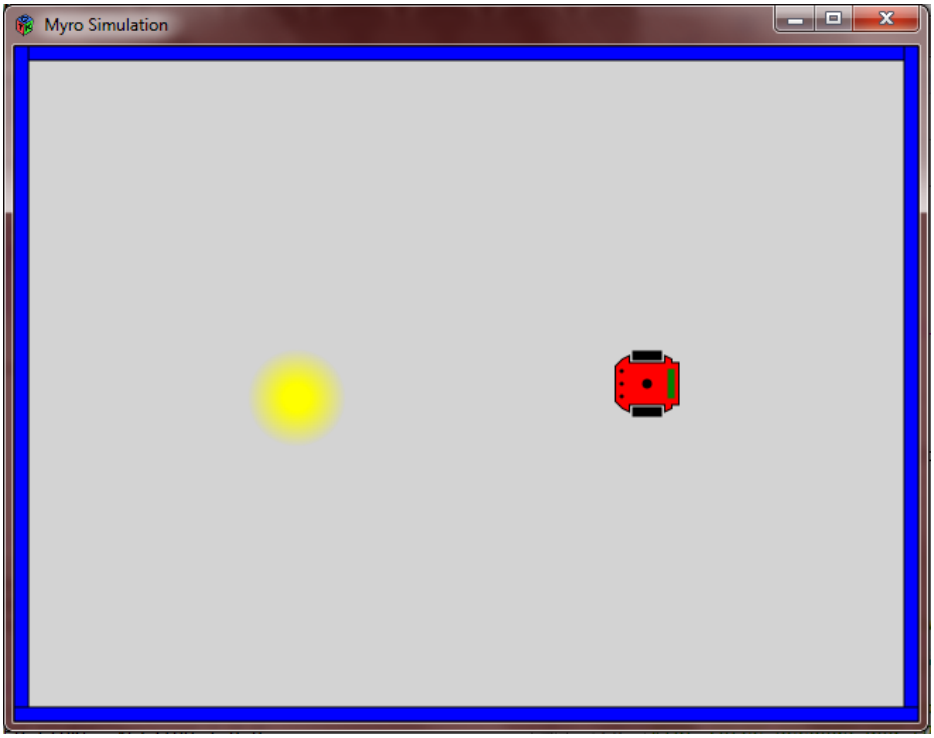


Figure 2.5. Scribbler simulator interface.

Python[9] is a widely used general-purpose[10], high-level[11] programming language. It is an object-oriented and structured programming language, meaning it makes extensive use of subroutines (functions/methods), block structures and loops[12]. Python also comes with a large and comprehensive standard library. Python was first started in the late 1980s, therefore there are many great tutorials for learning and mastering the language, one reason that it is perfect for

[9] http://www.python.org/

[10] General purpose means that it does not have a specific deign purpose in mind, unlike a language such as Structured Query Language, which is used solely for managing databases.

[11] "High-level language" refers to the higher level of abstraction from machine language. Languages like Java and C are both high-level languages as well.

[12] More information on these concepts can be found in the *Appendix* (see *Proposed Study Documents*).

beginners.[1] Since Python is a dynamically-typed programing language[13], it does not require declaring object and variable types, making it a bit easier for people just learning to program. It also has a plethora of additional libraries, or modules, that add or simplify features like game design[14], image manipulation[15], web development[16] and many others. Python is a great language for beginners and thus, an opportune language for the follow-up study, as well as IPRE's first choice for their studies.

2.4 IPRE Studies

Since IPRE believes that robots are an attractive way to introduce more students to computer science, they have conducted several studies that aim to investigate this hypothesis. [9] Summet et al (2009) details studies that IPRE ran in 2007 that involved teaching introductory computer science classes at three different schools. These classes enrolled 178 students and were taught using the personal Scribbler robots. Class sizes ranged from 12 to 104 students. The curriculum material includes a textbook, lecture notes and slides, hardware and software, and assignments. At the Georgia Institute of Technology, 90.97% (131 of 144 students) were successful (grade of A, B, or C) in the Fall 2007 Robots class, consisting of both CS majors and non-majors. The success rate in the Fall 2007 non-robots class, which consisted of non-majors, was 85.71% (78 of 91 students). The study found that on average the Robots students did 10% better than the Non-Robots students. The average annual enrollment in a Data Structures class at Bryn Mawr College from 1995 to 2006 was 7.45 students. The 2007 and 2008 classes, after the introduction

[13] Dynamically-typed programs verify the type safety of a program at runtime. C is a Statically-typed language, the opposite of Dynamic.

[14] http://www.pygame.org/news.html

[15] http://www.pythonware.com/products/pil/

[16] https://www.djangoproject.com/

of the introductory class with Robots, averaged 17.5 students, more than doubling the average enrollment. [23]

IPRE's approach attempts to provide interesting and diverse ranges of examples and exercises where the focus is on the context of the applications and not on the specific programming features one has to master. This challenges students in unique ways, and can lead students to pursue further studies in computing, which is one of the main goals of Hour of Code and IPRE.[12]

The studies that IPRE runs serve as the motivator for this project. First, as data to examine and second, as a stepping stone for a future study involving real robots and simulated robots, rather than robots versus non-robots.

3
RELATED WORK

This project spans several areas of study and there are many related articles of work. The main topics include that of teaching and robotics. This project began with an aim towards a younger audience than undergraduate level, thus many of the papers deal with teaching computer science to children, but many key principles still apply, such as interfaces, and key topics and principles.

3.1 Teaching

Scratch is a visual programming environment that allows users (primarily ages eight to sixteen) to learn computer programming while working on personally meaningful projects such as animated stories and games. A key design goal of Scratch is to support self-directed learning through tinkering and collaboration with peers, as well as introducing programming to those with no previous programming experience. Scratch uses a visual block language, a single-window user interface layout, and a minimal command set to simplify programming.[15] Scratch is a great programming tool for children trying to learn about computer science. While the IPRE studies and this senior project aim to be run with undergraduate-aged students, it was still very valuable

to read about what kinds of interfaces and problems work universally for teaching computer science to an audience who has not had contact before.

Tucker et al. (2003) proposes a model curriculum that can be used to integrate computer science fluency and competency throughout primary and secondary schools. This curriculum model provides a four-level framework for computer science, and contains roughly the equivalent of four half-year courses (many of these can be taught as modules, integrated among existing science and mathematics curriculum units). The first two levels suggest subject matter that ought to be mastered by all students, while the second two suggest topics that can be elected by students with special interest in computer science, whether they are college-bound or not.[25] Their curriculum acclimatizes students to using technology, which is certainly an exceedingly useful skill in today's technology-driven world; a world that will probably only get more technical. This was a very interesting paper to read, especially in regard to what topics they deemed the most important; universal topics such as the steps in algorithmic problem-solving, sequences, conditionals and loops (iteration).

Calico is another programming environment useful for new and experienced computer scientists alike, and is discussed in more detail in *Section 2.3*.[2] Calico was designed with an older audience in mind, and thus is the interface utilized in the proposed future studies of this project. Calico provides a great, easy-to-use interface, and comes equipped with Myro, allowing full control of the Scribbler robots, and the simulation interface.

As stated earlier, this project initially focused on a younger audience. In addition, while IPRE currently only engages with undergraduate students, they also aim to be able to engage with a wide range of ages. Therefore, it was worthwhile for this project to consider the worth of teaching computer science to children. Is it really worth teaching young kids to code? Some

individuals think not. Mark Guzdial, a Professor in the School of Interactive Computing at the Georgia Institute of Technology, is not convinced that you can "fruitfully teach five or six year olds to code—though it's certainly worth exploring and experimenting with." He says that the type of "coding" some young kids can learn is usually just sequences of statements. They rarely contain things like loops or conditionals—important aspects of programming. He also argues that teaching young kids the basics of programming has no effect long-term because the United States curriculums do not really teach any computer science, so any knowledge they might have acquired would be lost before they start learning computer science again at high school or college level.[7] If the United States had something like the Tucker et al. (2003) curriculum described above, perhaps it would be worthwhile. However, at this point, no such program is in place. Such arguments, and current lack of structure to facilitate young computer scientists, finalized the decision to focus this senior project on undergraduate students instead.

3.2 Robotics

Robots can be expensive, and are prone to noise, or errors, in their sensors and movements. They can be broken or lost, and can require maintenance to stay in top condition. However, they might be more engaging in a social context, and easier to physically influence during runtime than simulations. Visual programming systems such as Alice[17] or robot simulators such as Josef[18], Karel[19] and Logo Turtle graphics[20] still live "in the computer".[23] This means that you have to code in any changes you want to make to the robot or environment,

[17] http://www.alice.org/index.php

[18] http://machinarium.net/demo/

[19] https://www.cs.mtsu.edu/~untch/karel/

[20] http://el.media.mit.edu/logo-foundation/logo/turtle.html

and complicates making changes while a program is running. With real robots, you can interact with them and the environment directly.

The IPRE studies argue in favor of using robots to teach computer science, and several other studies seem to advocate their use. Correll et al. (2013) introduces a one-hour study for middle school aged children to engage in robotics and computer science. This paper studies using "Cubelets"[21], a modular robotic construction kit, which requires virtually no setup time and allows substantial engagement and change of perception of Science, Technology, Engineering and Mathematics (STEM) in as little as a 1-hour session. This paper describes the constructivist curriculum and provides qualitative and quantitative results on perception changes with respect to STEM and computer science in particular as a field of study. The curriculum was designed to teach students about several concepts at the core of computer science: decomposing problems into sub-problems and composing solutions to partial problems to solve a larger problem.[5]

Wyeth & Purchase (2013) use robots called Electronic Blocks[22], which function very much like the Cubelets. The blocks were made by placing electronics inside Lego Duplo Primo™ blocks. Electronic Blocks are a new programming interface designed for children aged between three and eight years. The programming environment includes sensor blocks, action blocks and logic blocks. By connecting these blocks, children can program structures that interact with the environment. The blocks provide young children with a programming environment that allows them to explore quite complex programming principles. The simple syntax of the blocks allows children to create and use simple code structures. The Electronic Block environment provides a developmentally appropriate environment for planning overall strategies for solving a problem, breaking a strategy down into manageable units, and

[21] https://www.modrobotics.com/cubelets

[22] http://itee.uq.edu.au/~peta/_ElectronicBlocks.htm

systematically determining the weakness of the solution. Electronic Blocks are the physical embodiment of computer programming. They have the unique dynamic and programmable properties of a computer minus its complexity.[26] It was amazing what these researchers could accomplish with such young children, and again iterated the important things to teach in beginning computer science: the strategies for breaking down and solving problems.

Martin (2007) explores challenges for both engineering and AI educators as robot toolkits evolve. He talks a lot about the importance of feedback in robotics, and that students need to get used to utilizing this feedback.[16] The feedback issue could be something that makes real robots better to use than simulated ones, an aspect of the proposed follow-up study of this project.

3.3 Robots as Motivators

Despite the results that IPRE found, other studies have concluded that robots have little effect over computer science attitudes and perceptions. The purpose of the study done by McGill (2012) was to determine whether using the Scribbler motivates students to learn programming in the form of a course with pre and post attitudes surveys, like the IPRE studies.

The study designed the surveys to study changes in opinions on attention, relevance, confidence and satisfaction, all parts of computer science attitudes and perceptions. The results of this study indicate that the use of these robots had a positive influence on participants' attitudes towards learning to program, but little or no effect on relevance, confidence, or satisfaction. Students were only slightly satisfied using the robots. This study indicates that other than increasing attention initially, there is no clear positive effect on motivation when using robots in an introductory programming course designed for non-majors. Robots in the classroom motivate students by sparking their interest, but do little else.[18]

3.4 An Hour versus a Semester

To recap, the Hour of Code is an opportunity for every student to try computer science for one hour. It has tutorials and ideas for people of all ages to try, that cover a wide variety of activities: game and app designing, robotics, creating cards and graphics, et cetera.

Hour of Code is not asking people to sign over their lives to computer science, they are simply asking people to try it. Only an hour of their time—a small commitment to many. Small enough that many people would agree to try it. However, is one hour long enough to change perceptions? Did people actually learn anything in one hour?

Hour of Code does have a "Beyond One Hour" section[23], filled with things to continue learning about computer science, so they obviously believed that some people would be interested enough to want to keep learning. In addition, if they were interested enough to keep learning, then their perceptions of computer science must have shifted towards the positive as well. Or at least they assume that. Corell et al. (2013) had hour-long sessions or studies in which they taught computer science, and also had positive results. Students showed a significant increase in interest to pursue computer science as a field of study after only an hour. [5]

Due to their design considerations and apparent successes, the proposed follow-up study is designed as hour-long coding workshops, as opposed to a semester-long course. Hour of Code also focused heavily on simulated or virtual applications; controlling virtual, simulated avatars (such as angry birds, snowmen, dogs, and of course, robots[24]), instead of having participants using real robots. The proposed follow-up design aims to study the pros and cons of this approach, and if Hour of Code could have been more (or less) successful with real robots, despite

[23] https://code.org/learn/beyond

[24] http://csedweek.org/learn

complications in accessibility the necessity of real robots creates. Thus, the goal of studying real robots versus simulated robots.

3.5 Robot Simulations

Robot simulations allow users to create applications for a robot without depending physically on the actual robot. Prensky (2013) classifies simulations into two "fidelity" categories based on how closely these simulated robots emulate their physical counterparts. There are "low fidelity" and "high fidelity" simulations. Low fidelity simulations are situations where one or a few elements are abstracted from reality to be emphasized. High fidelity simulations attempt to model reality as closely as possible.[20] The robot simulations that would be used in the follow-up study would be classified somewhere in between high fidelity and low fidelity. While the usage of the robot is high fidelity, the world that it exists in is not. The simulator looks different and is immune to noise (errors in the sensor readings/movements).

Robot simulations have also been used as a motivator for students to pursue computer science. The robot adventure game, outlined in Lee & Howard (2008), attempts to capitalize on the popularity of computer games to teach younger students basic computer science concepts and increase the students desire to pursue STEM-related careers in the future.[13] Mazur & Kuhrt (1997) provides a simulation platform for student creativity in programming, interaction between software modules, student cooperation and competition and classroom presentation of student efforts. They found that these extra additions can lead to greater student interest which inspires further discussion and learning.[17]

Simulations are much less expensive than robots, and have no noise error. They cannot be lost or broken, and do not require maintenance. You also do not need to have a separate copy for each student, unlike with real robots.[29] You can just download software and make it freely

available to a whole class. However, simulated robots could be less engaging in a social context, more isolated. It is also harder to physically influence the robot and its environment during runtime.

The results analyzed previously show that robots do not have much influence over attitudes, but perhaps simulations would be a better tool for changing perceptions and increasing learning/enjoyment of the students. The proposed follow-up study aspires to address these questions, and makes use of the Scribbler robots, also used in the IPRE studies.

4
STUDY ANALYSIS

Following the study in Summet et al (2009), IPRE ran additional studies at the University of Tennessee, Knoxville (UTK) in the Fall semester of 2009 and Spring semester of 2010. Like the Summet et al (2009) study, results were gathered from both Robotics classes and Non-Robotics classes.

4.1 Robotics Class

The Robots class was an introductory (100-level) computer science course that utilized robots. The course had no prerequisites and was open to all students. The course covered problem solving and algorithm development, organization and characteristics of modern digital computers with emphasis on software engineering, building abstractions with procedures and data, and programming in a modern computer language. It included design projects that required laboratory work, which focused on programming robots, specifically IPRE's Scribbler Robots.

The course was taught in the C++ programming language and follows the IPRE developed curriculum, as in Summet et al (2009). Students were asked to obtain these texts as

well: Learning Computing with Robots in C++, translated from Python by Deepak Kumar, and How to Think Like a Computer Scientist: Learning with C++, by Allen B. Downey.

It covered topics such as: basic programming concepts, C++ syntax and semantics, object-oriented programming, program development, sorting and searching, and robotics and AI. The full course syllabus can be found in the *Appendix* (see *IPRE Documents*).[11]

4.2 Non-Robotics Class

The Non-Robots class was also an introductory (100-level) computer science course. The course covered all of the same areas listed above, without utilizing robots to teach the concepts. This course also used the same How to Think Like a Computer Scientist: Learning with C++ textbook as the Robots class. The Non-Robots class also covered the same introductory topics as the Robots course. The courses were designed to be as similar as possible, with the only major difference being the utilization of the robot in the Robot class and the instructor teaching the course.[6]

4.3 Methodology

The data in this study were gathered as in the Summet et al (2009) studies. At the start of the semester, students were given a pretest survey to fill out to gauge initial computer science perceptions and attitudes. At the very end of the class, students were asked to fill out a posttest survey to study any changes made in attitudes and perceptions over the semester. The questions for all four survey conditions can be found in the *Appendix* (see *IPRE Documents*).

4.4 Results

All statistics were computed using the R Project for Statistical Computing[25]. Descriptive statistics were used to analyze the responses to the Likert scale questions. Likert responses were

[25] http://www.r-project.org/

coded into values ranging from one (strongly disagree) to five (strongly agree) for statistical testing. The Mann-Whitney's U test was used to evaluate relationships between the four groups: Pretest, Posttest, Robots and Non-Robots. The remainder of this section describes the results.

	Robots	**Non-Robots**
Pretest	Robots Pretest n = 22	Non-Robots Pretest n = 14
Posttest	Robots Posttest n = 27	Non-Robots Posttest n = 9

Figure 3.1. Punnett Square of study groups.

4.4.1 General Opinions and Observations

Mostly Computer Science majors and other sciences (Engineering, Physics, Math, etc.) enrolled in the two courses. Though, only two students rated themselves as being of intermediate programing experience. The rest were either beginner or had none. Students were also predominantly male and Caucasian. Out of all of the participants, only one had programmed a robot before.

Generally, students in the Non-Robots class ($\mu = 3.22$) seemed to enjoy themselves a bit more than those in the Robots class ($\mu = 3.07$) [Figure 4.8]. Students in the Robot class also liked being able to access the textbooks online ($\mu = 4.074$) instead of just having access to physical copies [Figure 4.13]. Both groups felt like they knew more about computers after completing the course than they did before taking them. [Figure 4.18]. However, in the Robots class, confidence in problem solving decreased slightly from Pretest to Posttest, while in Non-Robots it increased slightly [Figure 4.19].

Both Robots and Non-Robots also saw a decrease in seeking help for the courses between the Pretest and Posttest [Figure 4.25]. In Pretest and Posttest surveys, students in both groups

also indicated that they liked technology and regularly tried out new experiences and products [Figure 4.31] [Figure 4.32].

4.4.2 Statistically Significant Group Differences

Several of the questions had significant differences between groups. Results show that students in the Non-Robots ($\mu = 3.89$) course were more likely to write programs during the class that were not assignments than students in the Robots ($\mu = 2.89$) class ($p = .08$) [Figure 4.2]. Results also showed that students in the Robots class were much more likely to discuss assignments and lectures with peers who took the class ($\mu = 4.07$, $p = .01$) [Figure 4.6]. Robots students were also more likely to discuss the course material with peers who did not take the class ($\mu = 4.07$, $p = .10$) compared with students in the Non-Robots class ($\mu = 3.67$) [Figure 4.7].

Students in the Non-Robots class ($\mu = 4.38$) showed more enjoyment in being challenged by seemingly unsolvable situations or problems than the Robots ($\mu = 3.67$) class ($p = .04$), and showed an increase in enjoyment in being challenged between the Pretest ($\mu = 3.5$) and Posttest ($\mu = 4.38$) as well ($p = .02$) [Figure 4.22]. In the pretest, students in the Non-Robots class ($\mu = 4.14$) started out more likely to ask for help than the Robots ($\mu = 3.64$) class ($p = .09$) [Figure 4.26]. Additionally in the pretest, Non-Robots students ($\mu = 3.79$) started out thinking that other students in class knew more about computers than they did, than compared to the students in the Robots ($\mu = 3.14$) class ($p = .07$) [Figure 4.29].

4.4.3 Summary

Perhaps then, robots are not the answer. As found in McGill (2012), the robots seemed to have little effect over students' attitudes and perceptions of computer science. Nor did they seem to be a more enjoyable teaching method than standard computer science. Overall, students seemed to enjoy the Non-Robots class better. However, there were significantly less students in the

Non-Robots courses than the Robots courses, so that should be taken into account. Different instructors also taught the two courses, which could have influenced students' learning and enjoyment of the course. Another possibly influencing factor was the decision to teach both of the courses in C++, as compared to McGill (2012)'s choice of Python. Python is generally considered a much easier language for beginners.[22]

While there were significant differences in the Non-Robots, Pretest and Posttest groups, there were no significant differences found in the Robots group; that is, between the Robot Pretest and Robot Posttest. The only significant positive effect the robots seemed to have was that students were much more likely to discuss their assignments and lectures with other students, meaning the Robots class was more socially inclined than the Non-Robots. This is a particularly interesting result, and definitely positive in effect. If the goal of introductory computer science courses is to help spread the word about computer science (like the Hour of Code), than using robots certainly helps that cause, as people are more likely to speak about it to other people and increase the spread of awareness. Being more socially inclined, the robots also pose a beneficial tool to introductory classes, since new students can do well with a social support system behind them, something that many students feel lacking in when first beginning computer science.[18]

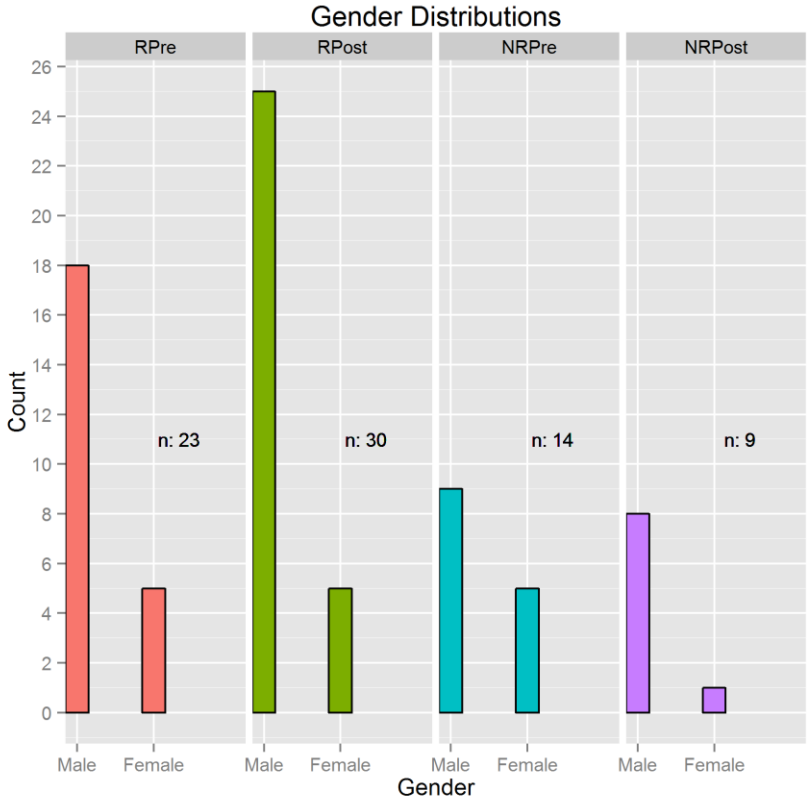


Figure 4.0. Gender Distribution of all groups.

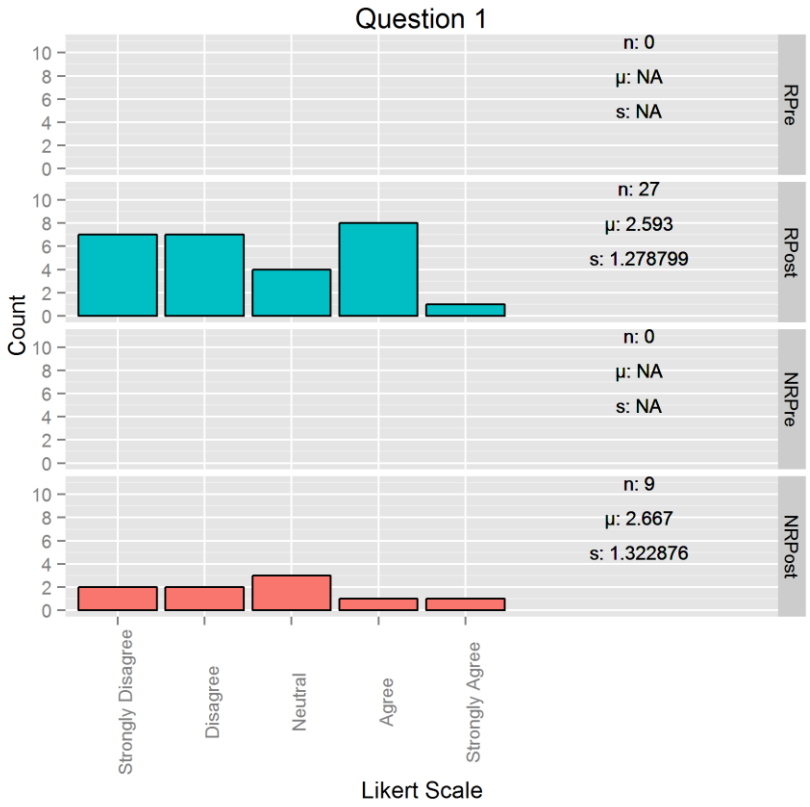


Figure 4.1. Survey Results for Question 1: “My experiences in this class caused me to decide to take another computer science class.”

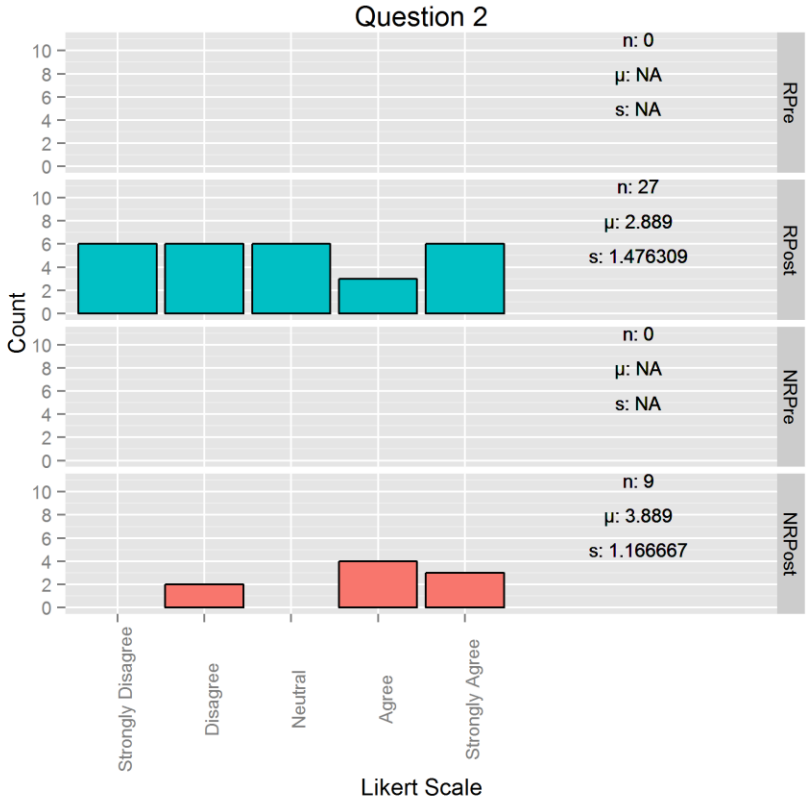


Figure 4.2. Survey Results for Question 2: "During the class, I wrote a program that was not an assignment for this class."

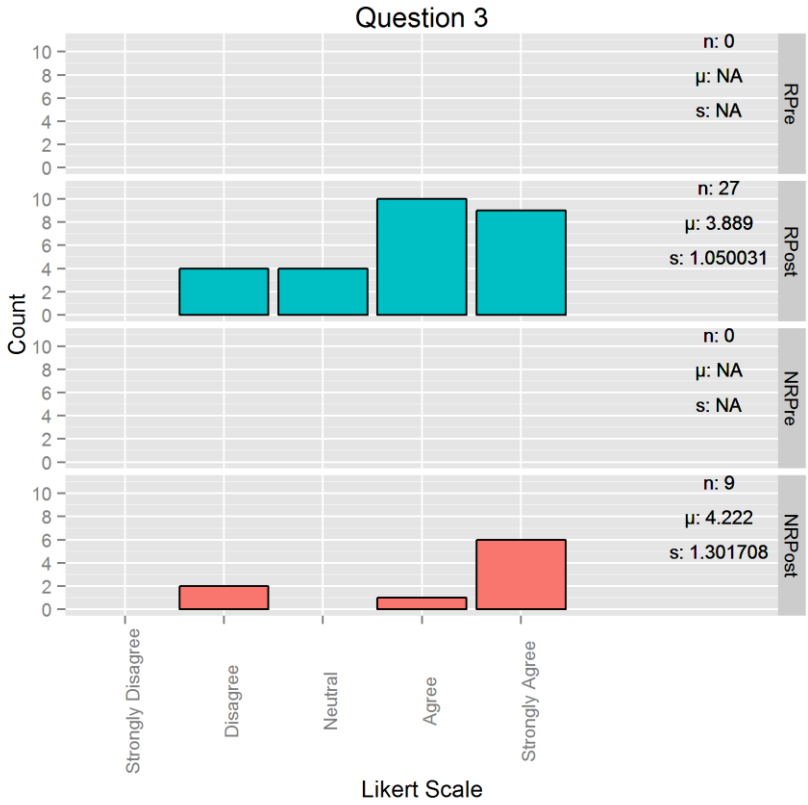


Figure 4.3. Survey Results for Question 3: "I expect that I will have to write a program (in any language) after I finish this class."

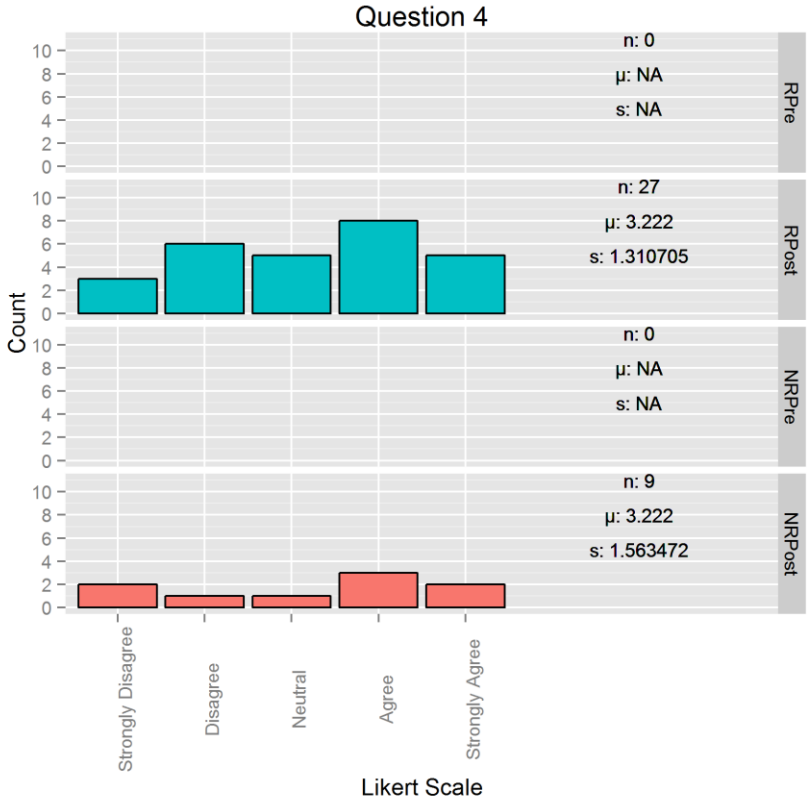


Figure 4.4. Survey Results for Question 4: "There was at least one homework that I spent extra time on because I thought it was cool."

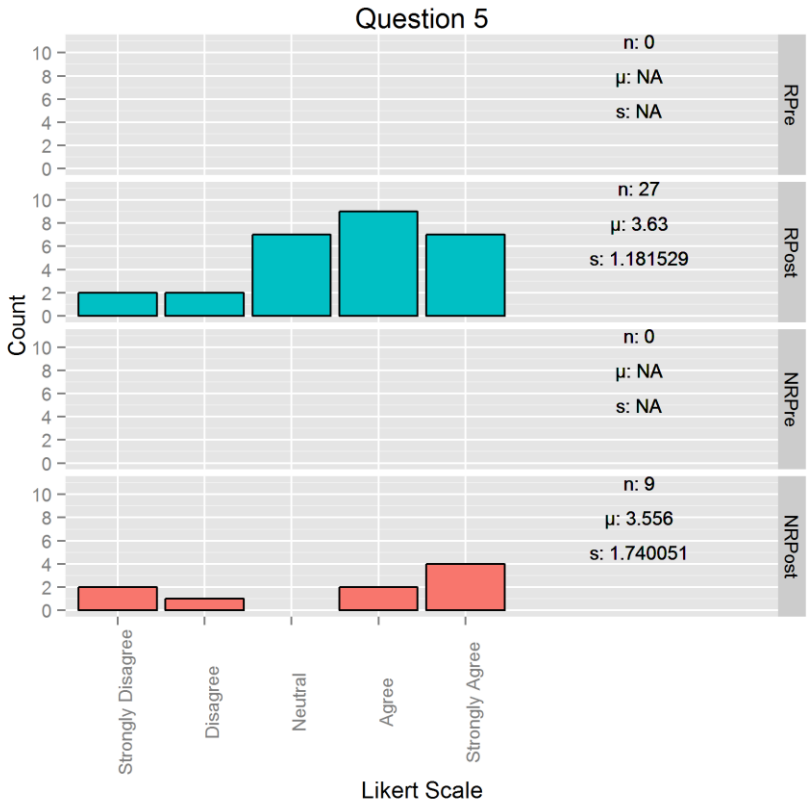


Figure 4.5. Survey Results for Question 5: "What I learned in this class is important to my future career."

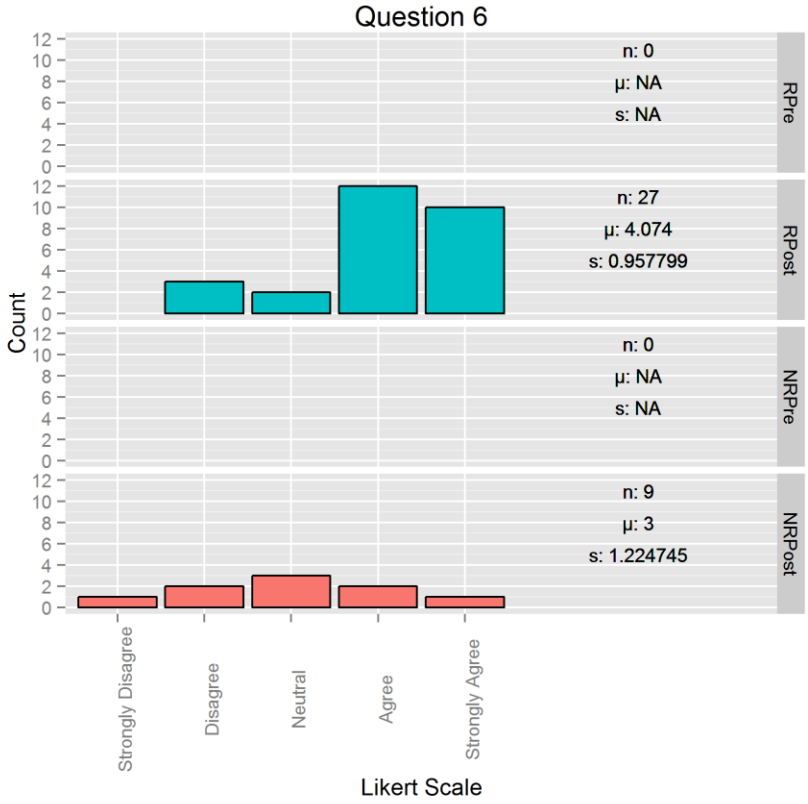


Figure 4.6. Survey Results for Question 6: "I discuss difficult assignments and/or detailed lectures with friends in the class."

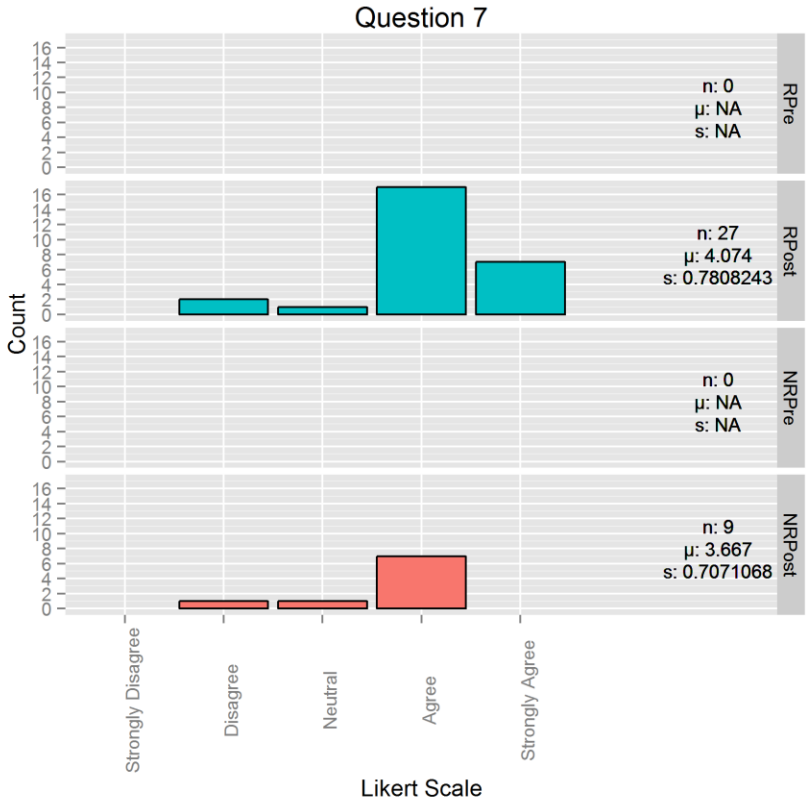


Figure 4.7. Survey Results for Question 7: "I talk with my friends (not in the class) about the class."

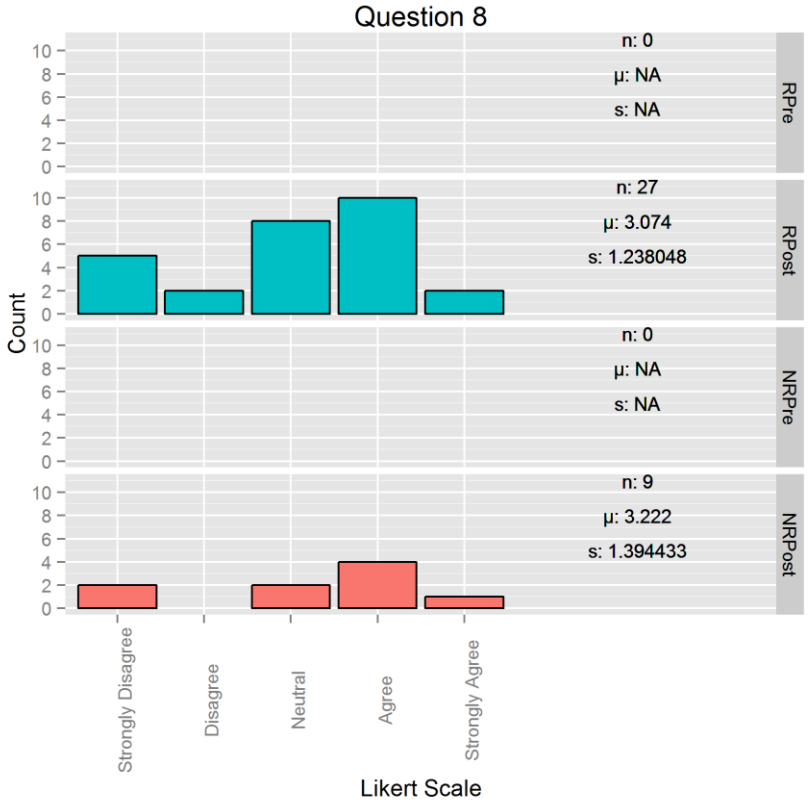


Figure 4.8. Survey Results for Question 8: "I enjoyed this class."

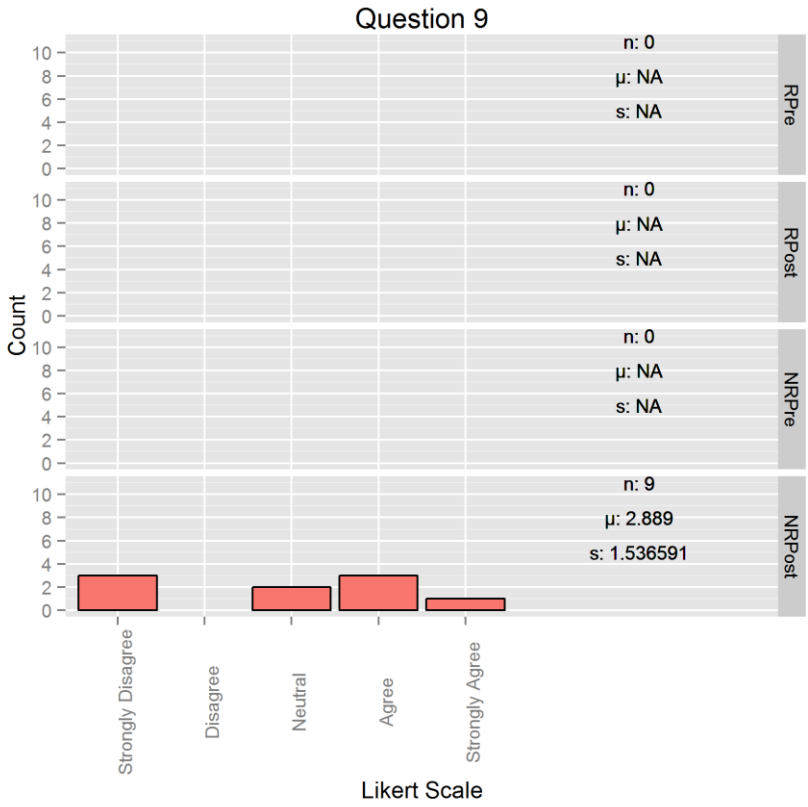


Figure 4.9. Survey Results for Question 9: "I would have liked to use robots in this class."

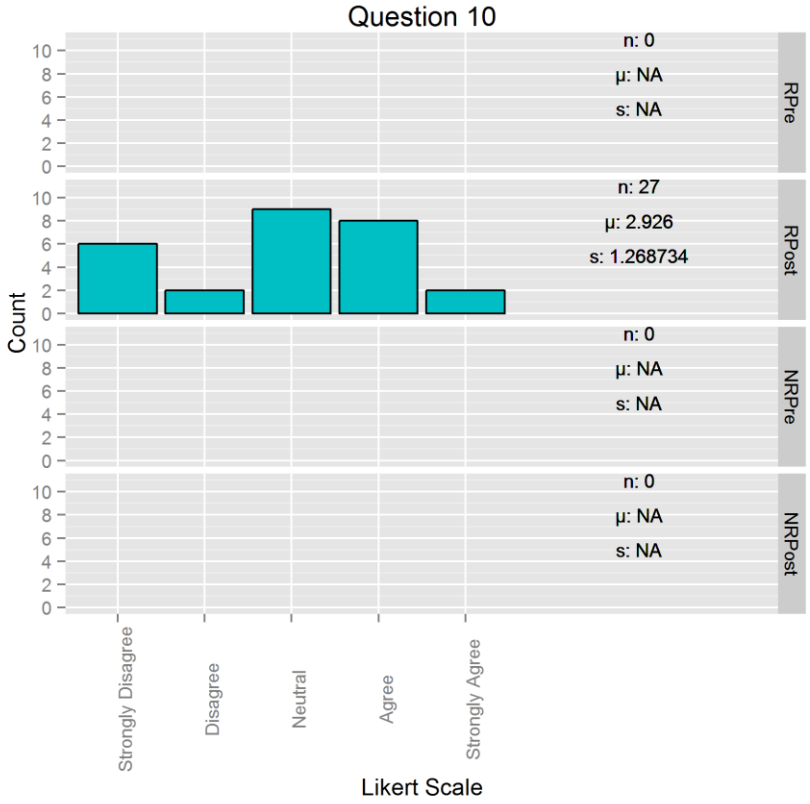


Figure 4.10. Survey Results for Question 10: "I enjoyed using the robot in this class.

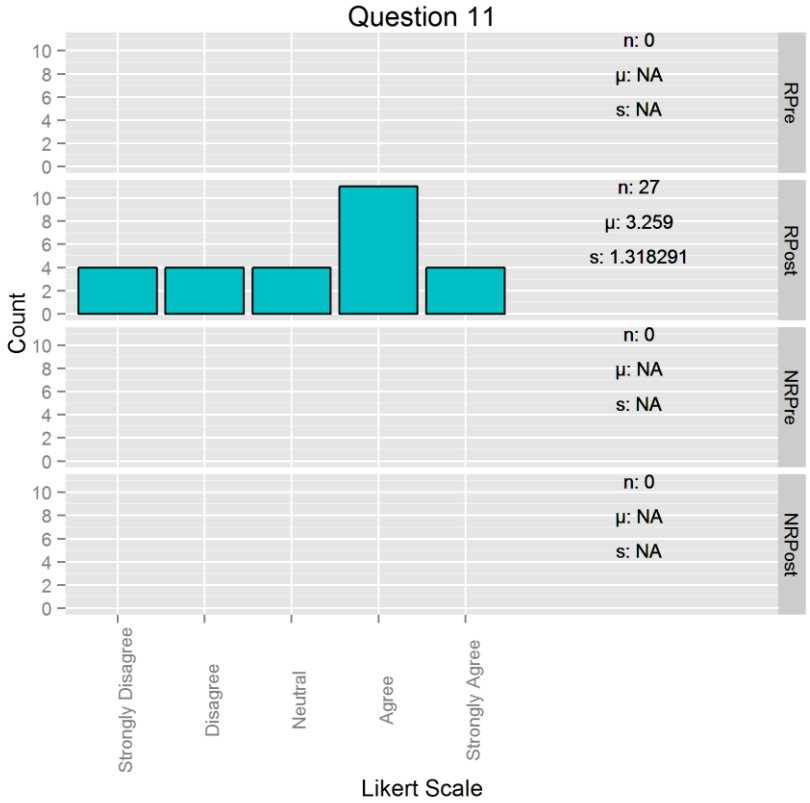


Figure 4.11. Survey Results for Question 11: "Maintaining the robot was easy."

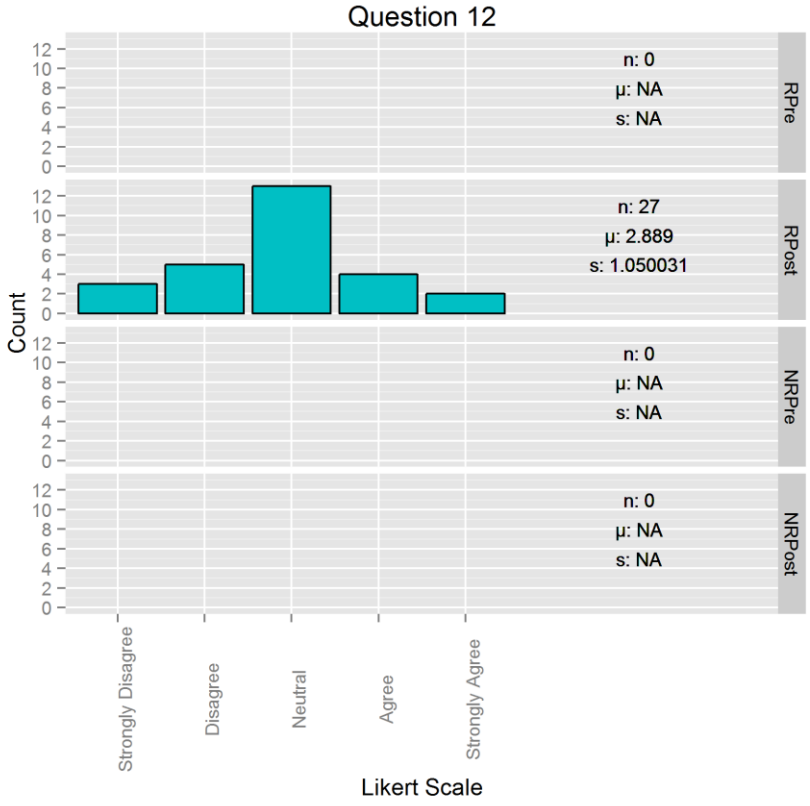


Figure 4.12. Survey Results for Question 12: "The pace of the lectures was too fast."

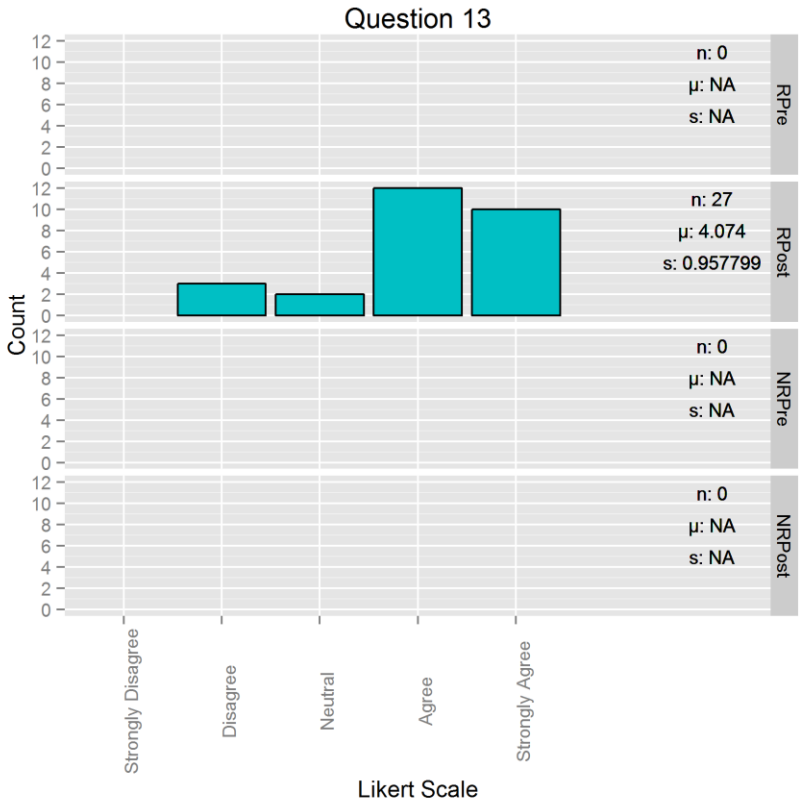


Figure 4.13. Survey Results for Question 13: "I liked being able to read the textbooks online."

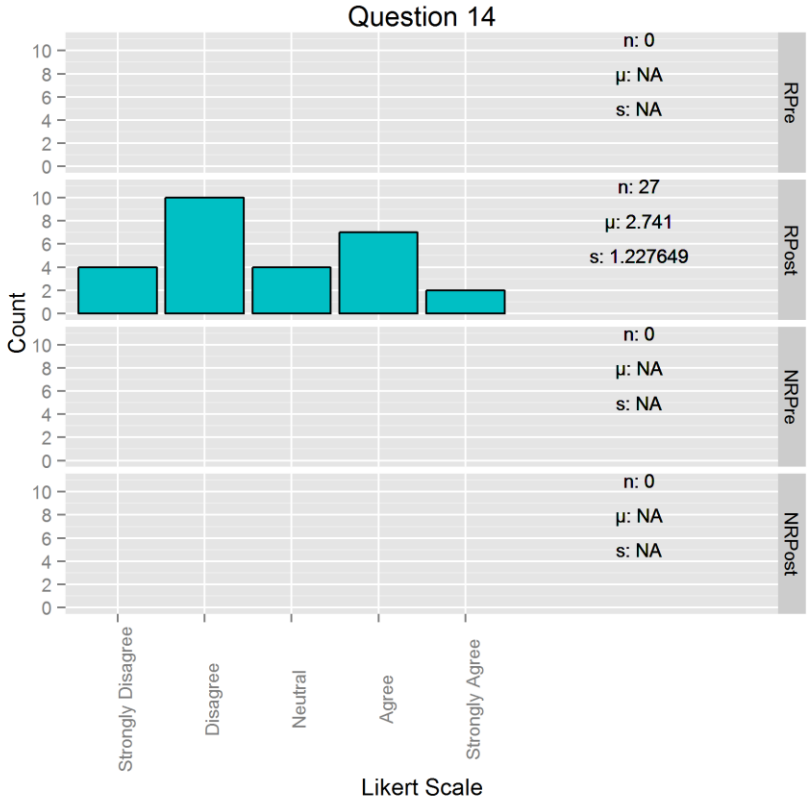


Figure 4.14. Survey Results for Question 14: "I did all the assigned readings."

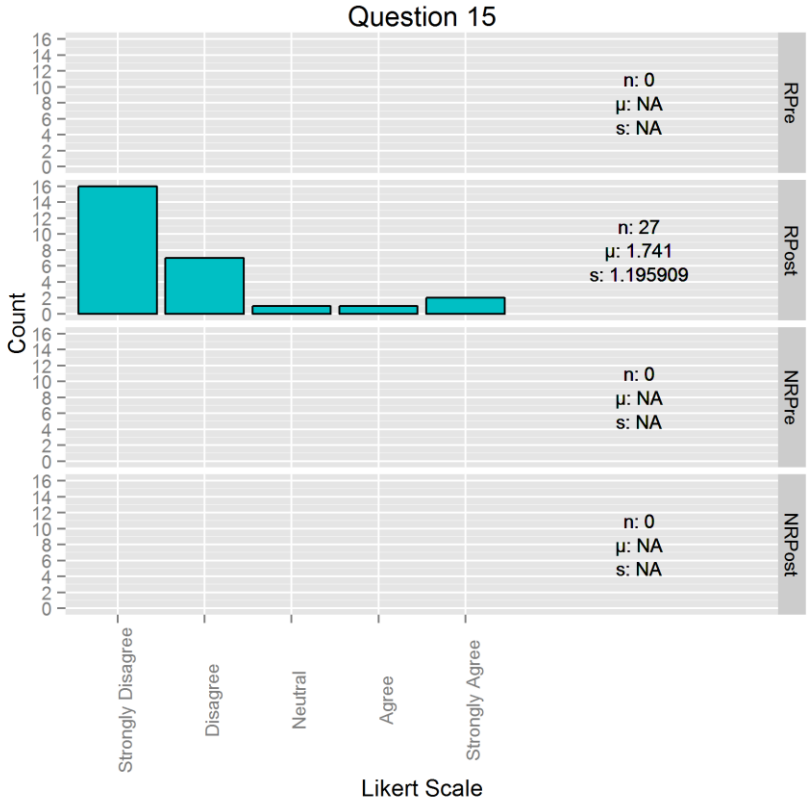


Figure 4.15. Survey Results for Question 15: "I purchased hardcopy of Learning Computing with Robots."

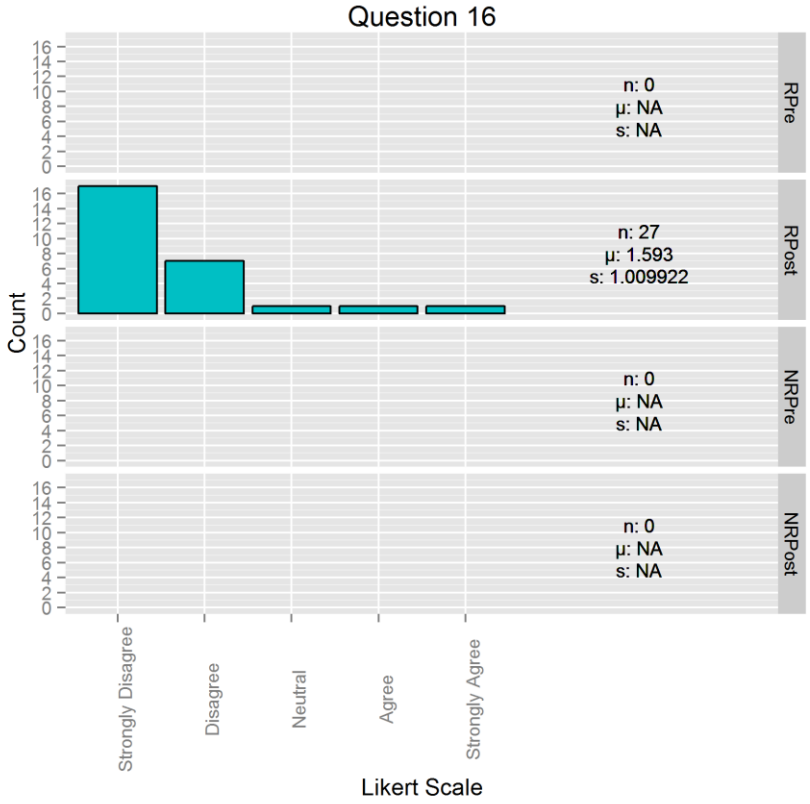


Figure 4.16. Survey Results for Question 16: "I purchased hardcopy of How to Think Like a Computer Scientist."

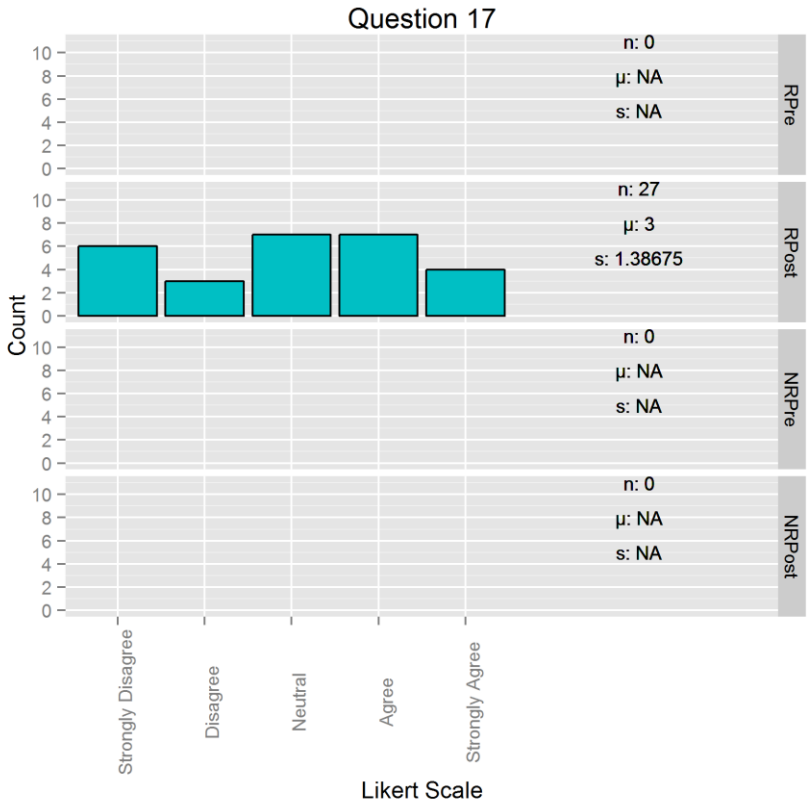


Figure 4.17. Survey Results for Question 17: "I enjoyed being able to take the robot home with me."

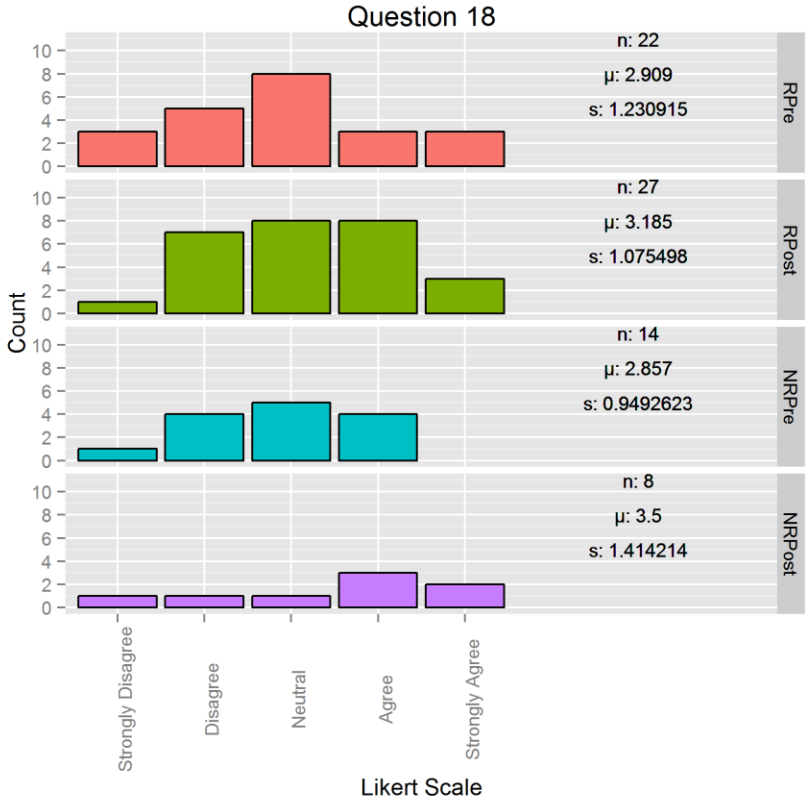


Figure 4.18. Survey Results for Question 18: "Compared to students in this class, I feel I know a lot about computers."

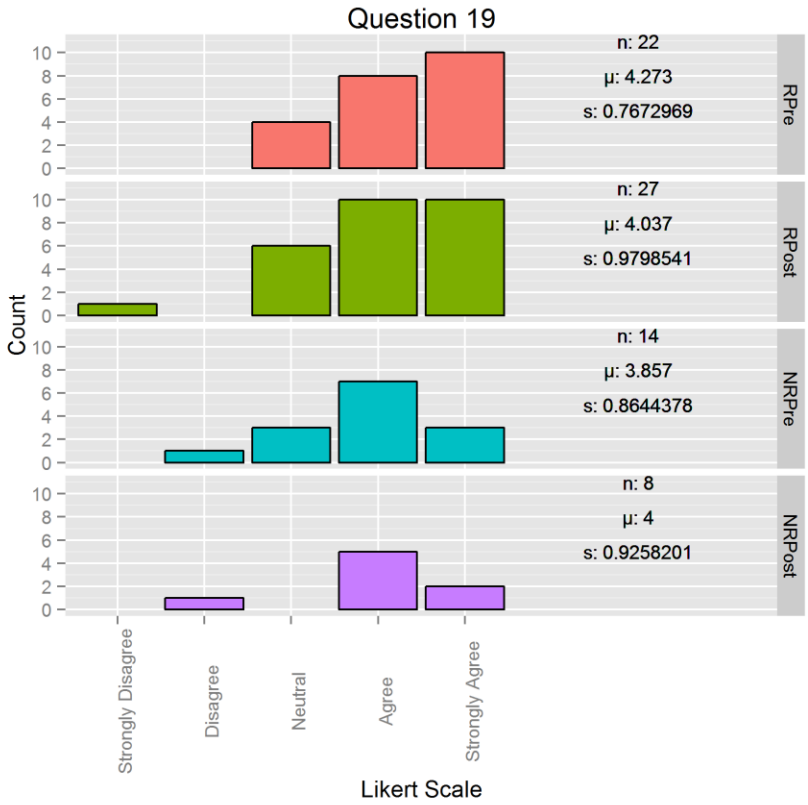


Figure 4.19. Survey Results for Question 19: "I am confident in my problem solving ability."

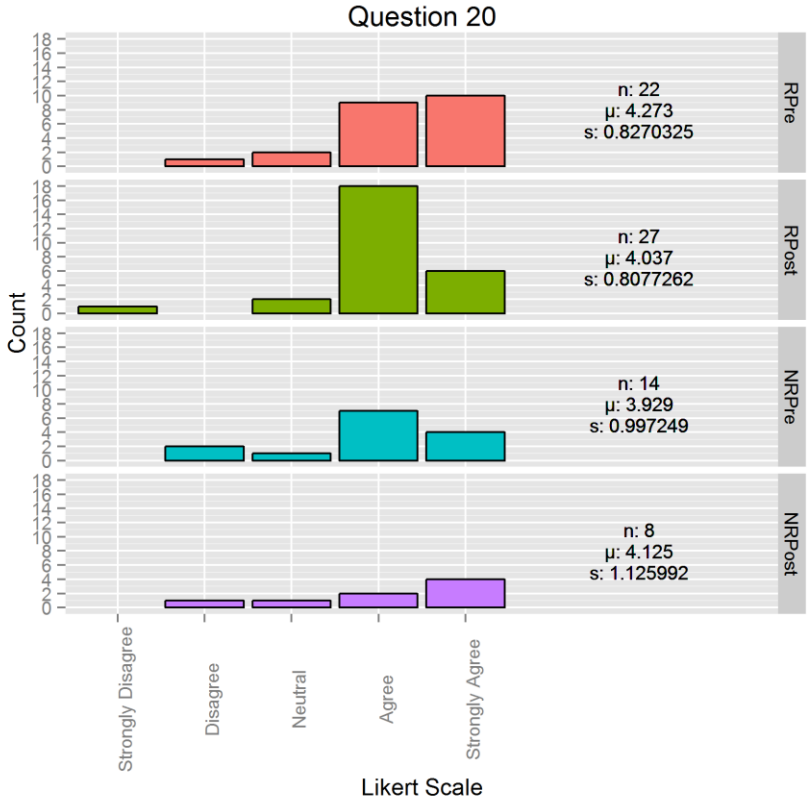


Figure 4.20. Survey Results for Question 20: "I am confident in my ability to persist until a solution is found."

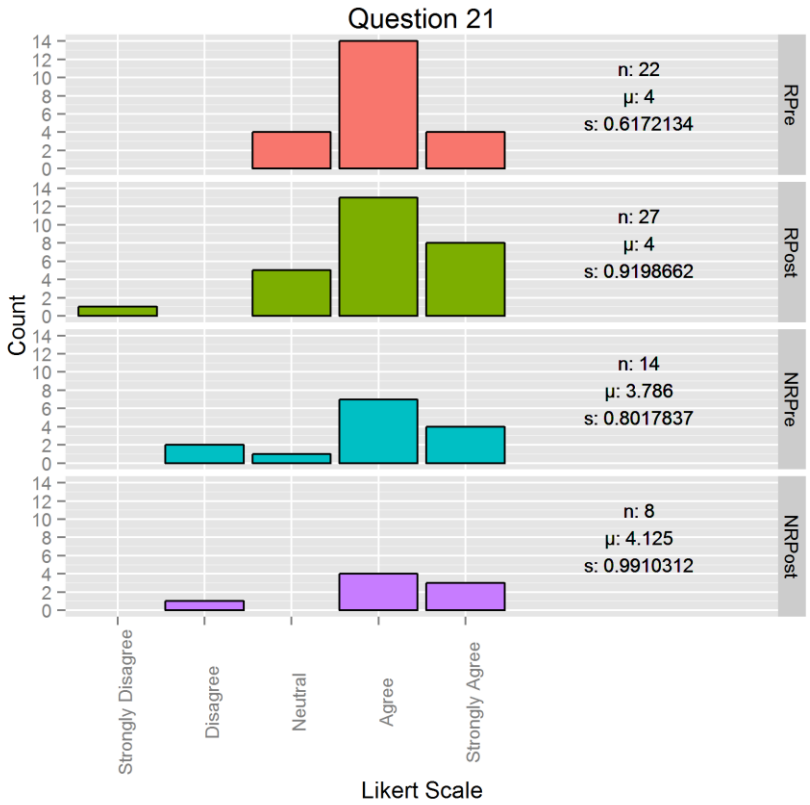


Figure 4.21. Survey Results for Question 21: "I am confident in my ability to meet unexpected challenges with success."

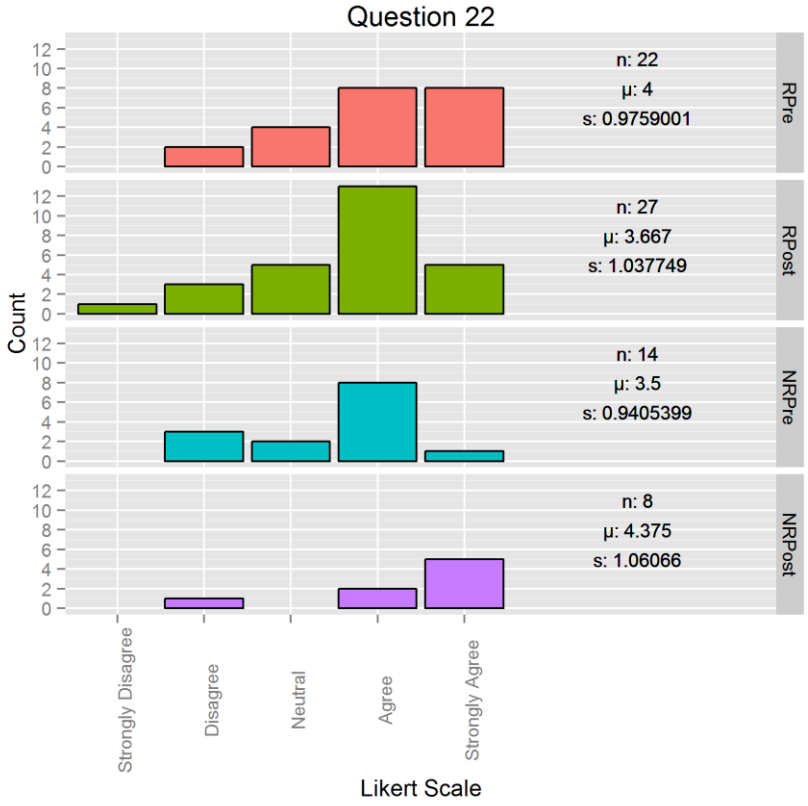


Figure 3.22. Survey Results for Question 22: "I enjoy being challenged by seemingly unsolvable situations or problems."

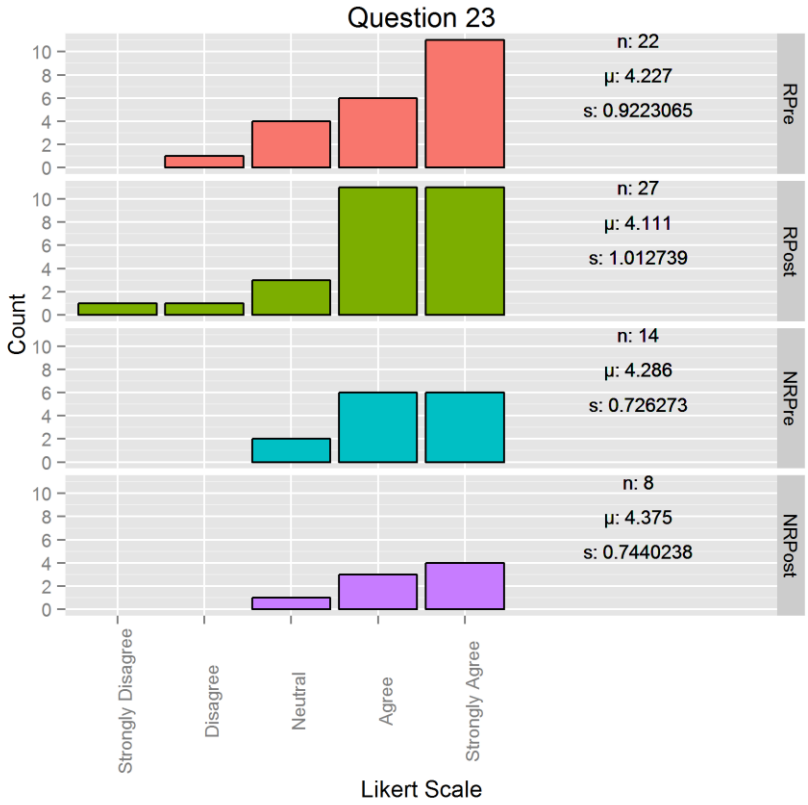


Figure 4.23. Survey Results for Question 23: "I am confident in my math ability."

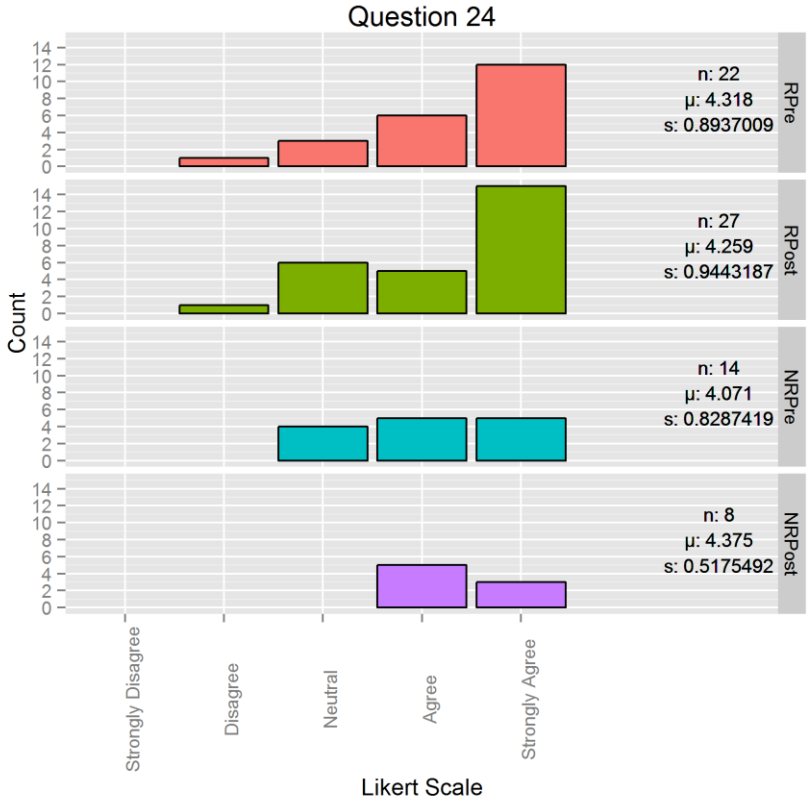


Figure 4.24. Survey Results for Question 24: I am confident in my science reasoning ability."

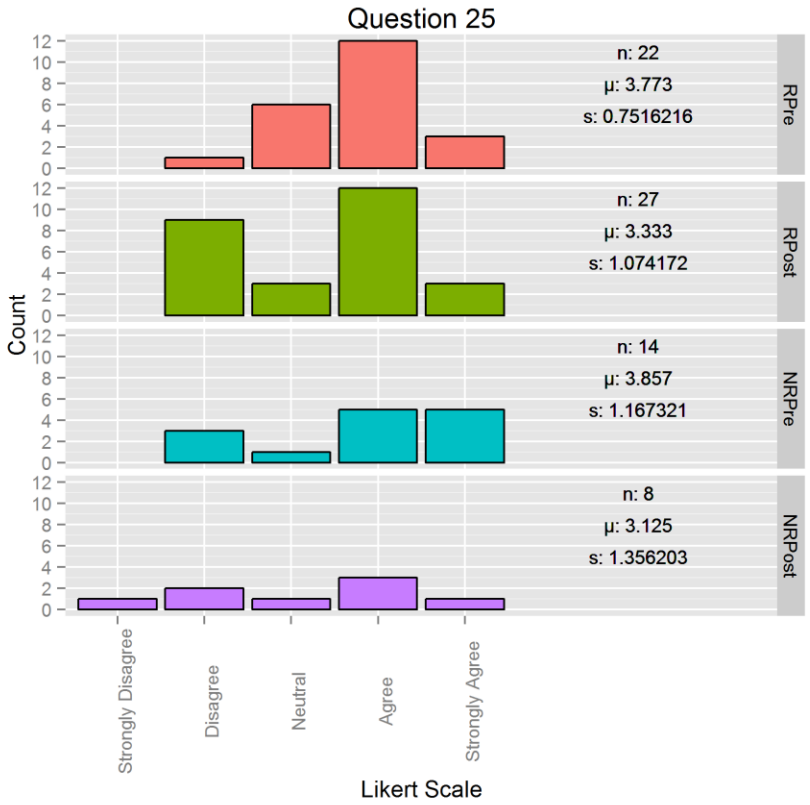


Figure 4.25. Survey Results for Question 25: "I seek help in class before I become very frustrated."

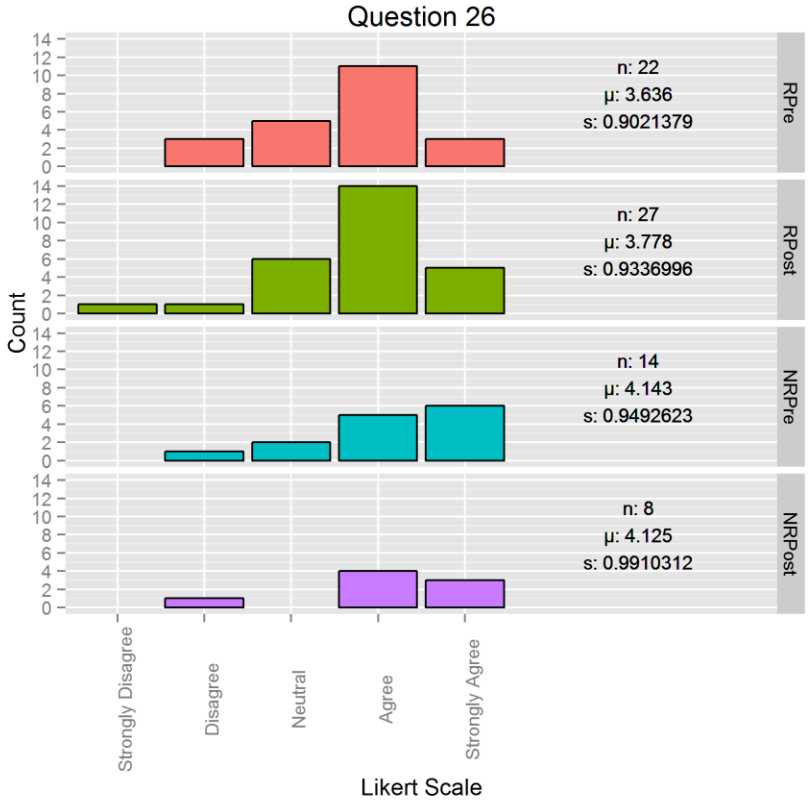


Figure 4.26. Survey Results for Question 26: "I am not afraid to ask for help."

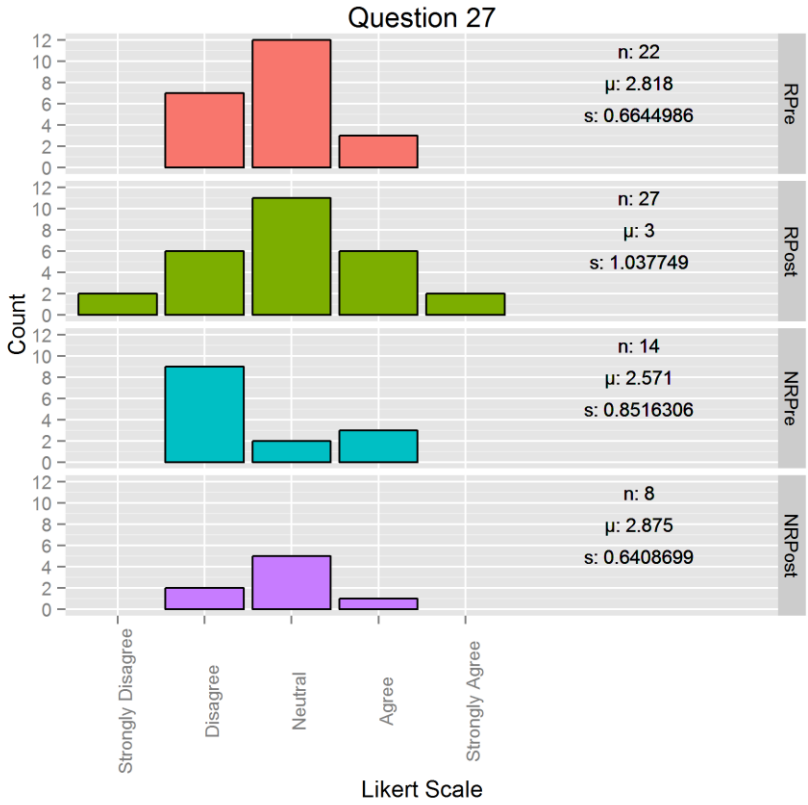


Figure 4.27. Survey Results for Question 27: "Computer Science and programming are the same thing."

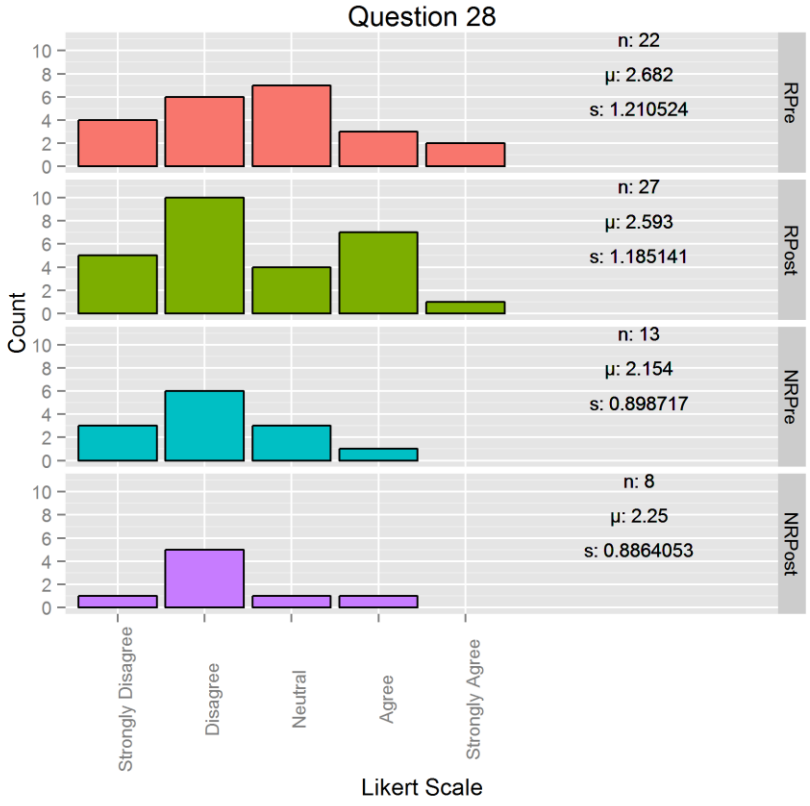


Figure 4.28. Survey Results for Question 28: "I took this course to see what computer science is all about."

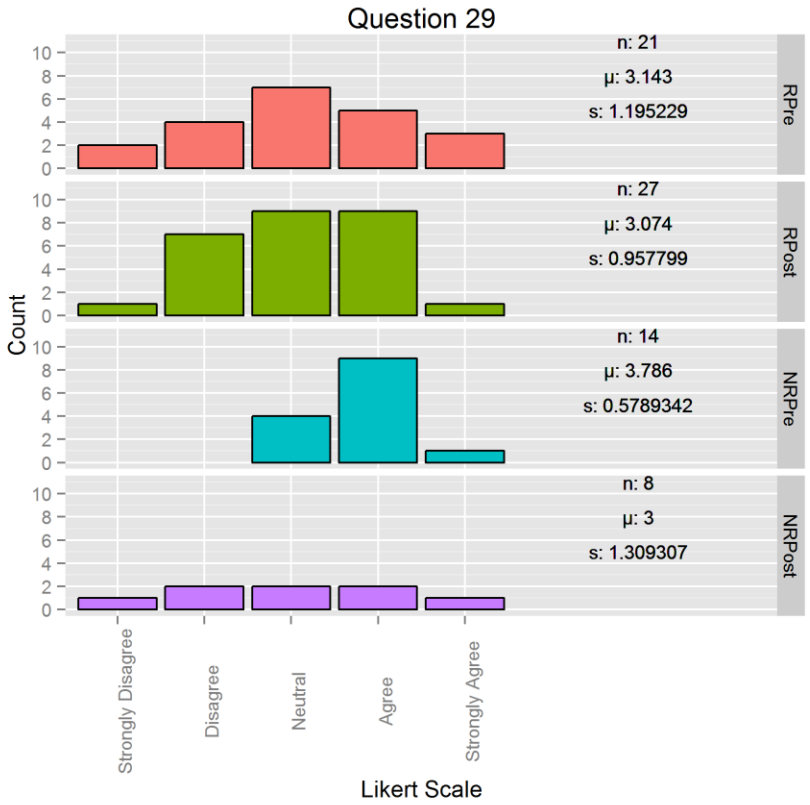


Figure 4.29. Survey Results for Question 29: "Other students in class know more about computers than I do."

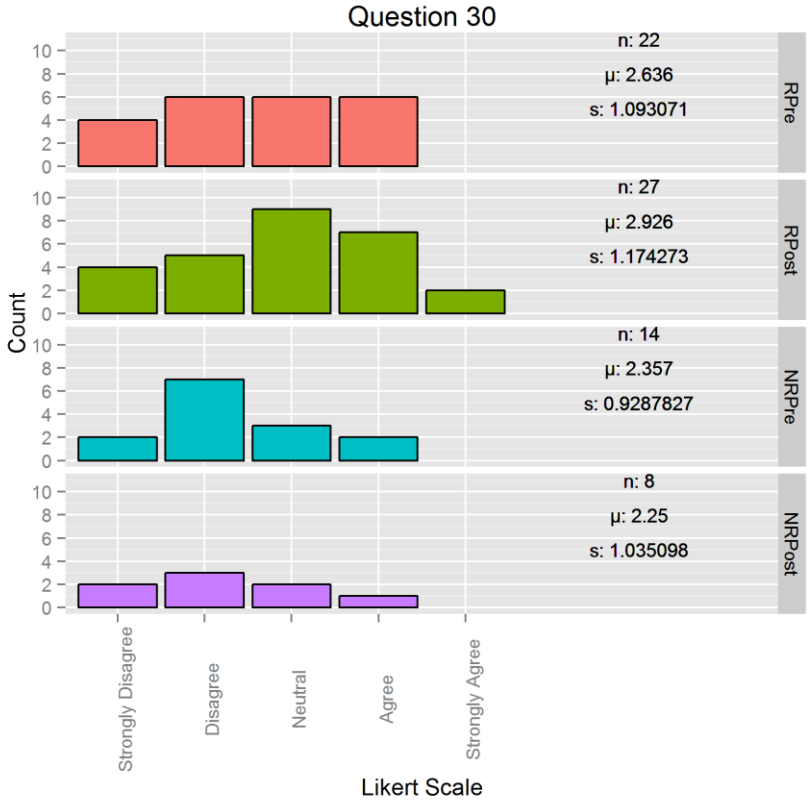


Figure 4.30. Survey Results for Question 30: "I dislike situations or problems that are seemingly unsolvable."

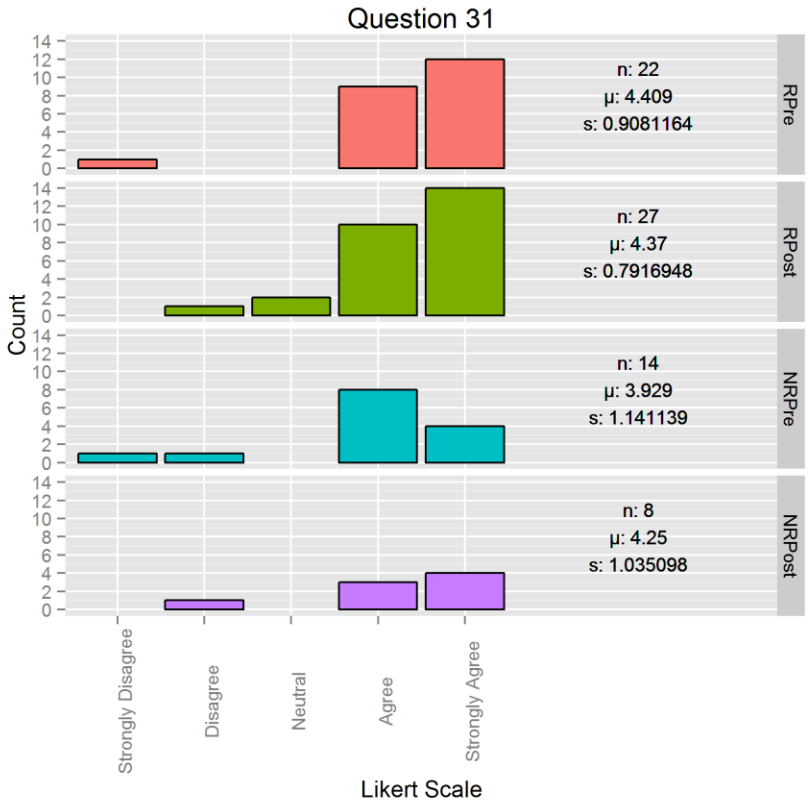


Figure 4.31. Survey Results for Question 31: "I like technology."

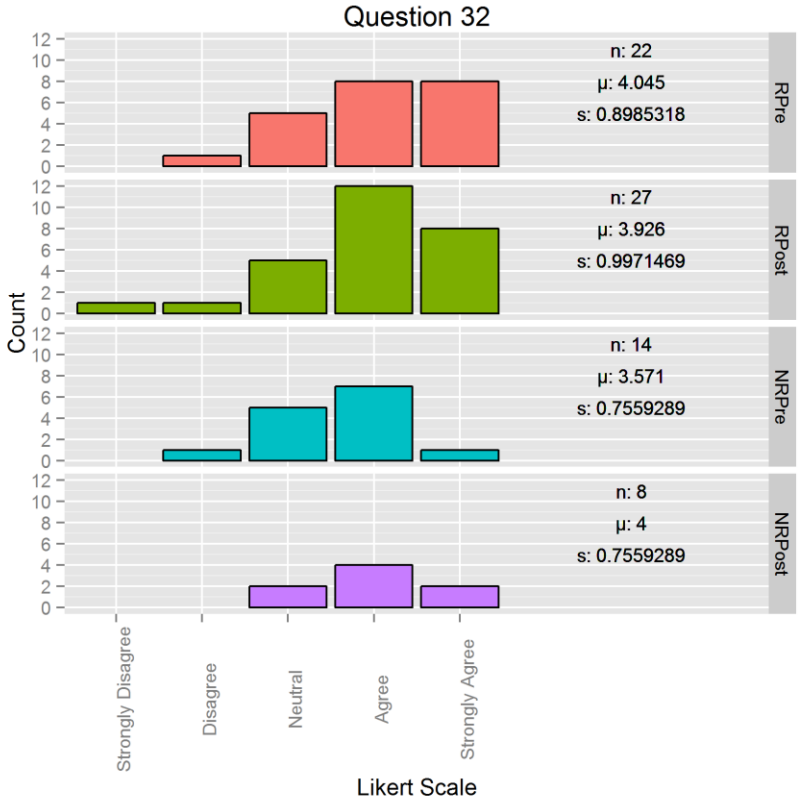


Figure 4.32. Survey Results for Question 32: "I regularly try out new experiences and products."

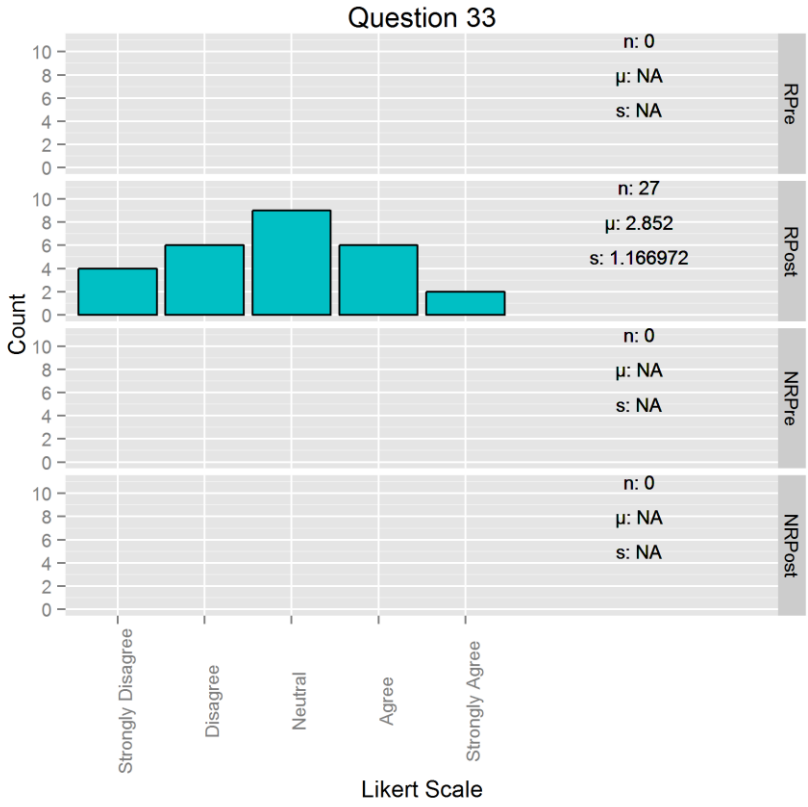


Figure 4.33. Survey Results for Question 33: "I am confident programming in C++."

5
MOVING FORWARD

This project proposes a design and implementation of computer science coding workshops that would focus on changing computer science attitudes and perceptions of its participants. The coding workshops would be an hour-long introduction to Calico, Python and controlling a Scribbler robot—either real or virtual, through the use of the Myro simulator. In this hour of time, participants would work on two programming tasks, each of which can be graded in terms of correctness, designed to make use of common programming principles and tactics. By the end of the workshop, participants would hopefully have a basic understanding of Python, Calico and several general programming concepts. This study aims to understand the differences between using real robots to introduce computer science as opposed to using virtual, simulated robots as an introduction to computer science. It also aspires to gauge whether computer science attitudes and perceptions could be changed within the span of an hour-long workshop, instead of over an entire semester.

5.1 Methodology

Testing participants would be solicited from within the Bard community. Although the test pool would be limited to current Bard students at least eighteen years of age and people with little or no programing background, no effort would be made to specifically target participants from certain demographics. The expectation is that the final group of participants would contain members from mixed demographics.

Time (minutes)	Activity
0-15	Consent Forms and Pretest Survey
15-25	Introduction
25-45	Task One: Dance Routine
45-75	Task Two: Light Finder
75-90	Posttest Survey

Figure 5.1. Study Schedule

Each participant would be asked to fill out a Consent Form and a Pretest Survey, which aims to garner what kind of, if any, computer science background each participant has had, and to establish their current views of computer science and technology. Participants would then be paired up with a partner to participate in the workshop. The workshop would be partitioned into three sections: the introduction, first task and second task. After the workshop, each participant would be asked to fill out a Posttest Survey, which aims to assess whether participants original perceptions of computer science changed at all during the workshop. Depending on which trial the participants are in, there would also be questions asking about their experience with either the simulations or real robots, and if they thought the opposite would have been more interesting or engaging. The entire process was designed to take an hour and a half; an hour for the workshop, and fifteen minutes before and after to fill out the surveys and consent form. There are two different conditions: a real robot trial and a simulator robot trial.

5.2 Introduction

The introduction section is designed to take ten minutes. Examples would be shown either using the simulator or a real robot, depending on which trial condition the group is in. It follows a series of PowerPoint slides, which illustrate the basics of the Calico program and Python. The next slides cover basic commands that participants would be using in the workshop, and how to use them. These slides are followed by an introduction to Loops and Conditionals. The last part of the introduction details the two programming tasks that the participants would be asked to complete, and the introduction ends. The participants would then begin the first task.

5.3 First Task

The participants would be given twenty minutes to work on the first task. The first programming task is designed to familiarize participants with using Calico and Python. Their task is to create a simple dance sequence for their robot, which should last at least twenty seconds. This task was designed to be easy and fun, but also introduce participants to the programming concepts of sequences and program structure, and encourage them to make use of loops. A solution that runs for twenty seconds using a loop structure is graded as a success condition. Partial success is given for programs that use basic sequential commands that add up to twenty seconds worth of behavior.

5.4 Second Task

The participants would be given thirty minutes to work on the second task. The second programming task is designed to introduce participants to higher-level concepts and breaking down problems. Their task is to make the robot drive around and try to locate a light source in the environment. Once it does, the robot should beep to let the participant know the light has been found. This task is designed to be a bit more difficult, and makes use of more aspects of the

robot, besides movement. Participants would have to use the light sensors, beep function, relational operators (>, <, ==, !=), the Boolean data type, and would be encouraged to use both loops and conditionals to solve this task. A solution that finds the light, and uses both loops and conditionals would be graded as a complete success condition. Partial success would be given for using either loops or conditionals but not both.

All study materials can be found in the *Appendix* (see *Proposed Study Documents*).

6
CONCLUSION

This project has presented an examination and analysis of the Institute for Personal Robots in Education studies of 2009 and 2010 that judged students attitudes of computer science before and after an introductory computer science course. The course was taught in one of two ways, with robots or without them. The results from these studies show that the robots did not seem to have any significant effects on computer science perceptions. However, the robots did increase the likelihood that students discussed their assignments and lectures with their peers, a factor that could lead to increasing involvement in computer science.

This project then detailed a follow-up study that could be run as an extension of the IPRE studies, examining the effects of using real robots versus simulated robots. This study is designed to address two research questions. The differences in robots and simulated robots on changing computer science perceptions, and if an hour-long workshop is enough time to change said perceptions.

Many complaints from McGill (2012) and the IPRE study short answer results were about the robot hardware—batteries, not being calibrated correctly, not driving straight, faulty

sensors—the list goes on. Perhaps the aversion to the robots has less to do with a lack of interest in the subject, and rather the irritations of additional—sometimes faulty—hardware. If this is the case, simulated robots could be the happy medium of the two—Robots without hardware upkeep. In order to tell if students would still just prefer no robots at all—simulated or not—another test would have to be done comparing the utilization of simulated robots and no robots in general, assuming that is, that there are some differences between using robots and simulated robots.

APPENDIX

The following pages present the documents, tables and figures that compose the methodology and results of the IPRE study and proposed follow-up study.

IPRE Results

Question Key

Q1	My experiences in this class caused me to decide to take another computer science class.
Q2	During the class, I wrote a program that was not an assignment for this class.
Q3	I expect that I will have to write a program (in any language) after I finish this class.
Q4	There was at least one homework that I spent extra time on because I thought it was cool.
Q5	What I learned in this class is important to my future career.
Q6	I discuss difficult assignments and/or detailed lectures with friends in the class.
Q7	I talk with my friends (not in the class) about the class.
Q8	I enjoyed this class.
Q9	I would have liked to use robots in this class.
Q10	I enjoyed using the robot in this class.
Q11	Maintaining the robot was easy.
Q12	The pace of the lectures was too fast.
Q13	I liked being able to read the textbooks online.
Q14	I did all the assigned readings.
Q15	I purchased hardcopy of Learning Computing with Robots.
Q16	I purchased hardcopy of How to Think Like a Computer Scientist.
Q17	I enjoyed being able to take the robot home with me.
Q18	Compared to students in this class, I feel I know a lot about computers.
Q19	I am confident in my problem solving ability.
Q20	I am confident in my ability to persist until a solution is found.
Q21	I am confident in my ability to meet unexpected challenges with success.
Q22	I enjoy being challenged by seemingly unsolvable situations or problems.
Q23	I am confident in my math ability.
Q24	I am confident in my science reasoning ability.
Q25	I seek help in class before I become very frustrated.
Q26	I am not afraid to ask for help.
Q27	Computer Science and programming are the same thing.
Q28	I took this course to see what computer science is all about.
Q29	Other students in class know more about computers than I do.
Q30	I dislike situations or problems that are seemingly unsolvable.
Q31	I like technology.
Q32	I regularly try out new experiences and products.
Q33	I am confident programming in C++

Tables

Robot Pretest Survey Results

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
																	1	3	3	3	2	5	4	3	4	3	1	5	4	4	2	
																	3	4	4	4	3	4	4	4	4	3	3	3	3	4	4	
																	2	3	2	3	3	3	2	3	2	3	3	4	4	4	3	
																	5	5	5	5	5	5	5	5	5	3	5	1	3	5	5	
																	4	4	4	4	4	2	3	3	4	4	4	2	3	5	5	
																	3	5	4	3	3	3	5	3	3	3	5	4	3	5	5	
																	5	5	5	5	5	4	4	4	3	2	1		2	5	5	
																	4	5	5	5	5	5	4	4	4	4	2	1	2	5	5	
																	2	4	5	4	5	3	3	5	5	2	3	4	1	5	5	
																	3	3	3	3	4	4	3	4	4	2	4	5	4	4	4	
																	2	5	4	4	4	5	4	5	4	4	2	4	2	1	3	
																	1	4	4	4	3	5	4	4	4	3	4	2	3	4	4	
																	5	5	5	5	5	5	5	5	3	3	3	3	1	5	5	
																	3	5	4	4	4	3	5	4	2	3	2	3	3	5	4	
																	1	4	4	4	4	5	5	5	5	3	1	5	1	4	4	
																	3	5	5	4	5	4	4	4	4	3	3	3	1	4	3	
																	2	4	5	4	4	5	5	4	3	2	2	3	4	4	3	
																	3	4	5	4	4	5	5	5	4	4	2	3	4	4	3	
																	4	5	4	4	4	5	5	5	4	3	3	2	2	5	4	
																	3	5	4	4	5	5	5	5	4	4	3	2	2	5	4	
																	3	4	5	4	5	4	4	4	4	2	2	3	2	5	5	
																	2	3	4	4	2	4	4	2	2	3	4	3	4	5	4	

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
Min																			1	3	2	3	2	2	2	2	2	1	1	1	1	2	
1st Qu																			2	4	4	4	3.25	4	4	3	3	2	2	2	4	3.25	
Median																			3	4	4	4	4	4.5	5	4	4	3	3	3	5	4	
Mean																			2.909	4.273	4.273	4	4	4.227	4.318	3.773	3.636	2.818	2.682	3.143	2.636	4.409	4.045	
3rd Qu																			3.75	5	5	4	5	5	5	4	4	3	3	4	3.75	5	5	
Max																			5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	
SD																			1.230915	0.7672969	0.8270325	0.6172134	0.9759001	0.9223065	0.8937009	0.7516216	0.9021379	0.6644986	1.210524	1.195229	1.093071	0.9081164	0.8985318	
Responses	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		22	22	22	22	22	22	22	22	22	22	21	22	22	22	0

Robot Posttest Survey Results

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
4	5	5	5	5	4	4	3		1	4	3	5	5	5	1	1	3	4	4	4	4	4	4	4	4	4	3	2	1	2	5	5	5
2	3	3	3	4	4	2	3		3	4	2	4	2	2	2	2	4	2	3	4	3	3	4	3	2	3	3	2	4	4	4	3	2
4	2	2	4	5	4	4	4		4	4	3	4	2	3	3	3	4	3	4	4	4	4	4	4	4	4	3	4	4	4	5	4	3
2	2	4	3	4	5	5	4		4	4	3	4	2	2	2	3	3	3	4	4	4	4	5	5	4	4	3	3	3	3	5	5	3
1	1	2	1	1	5	4	1		1	2	3	5	5	1	1	1	2	4	4	4	4	3	4	3	4	5	1	1	3	3	4	3	1
3	5	5	3	4	2	4	3		3	4	3	4	4	1	1	3	2	3	4	4	4	3	3	3	2	2	3	5	4	3	4	3	2
4	5	5	2	5	5	4	4		1	1	1	4	1	1	1	1	4	2	5	4	5	4	5	5	5	5	4	1	2	3	5	4	4
4	2	4	4	4	4	4	4		4	4	4	5	3	1	1	1	4	2	3	4	4	4	4	4	2	3	4	4	4	3	5	5	3
3	3	3	2	4	3	3	3		3	4	3	3	2	2	2	3	3	3	3	4	4	4	2	3	4	4	3	2	3	4	3	3	2
2	5	5	5	5	5	5	4		1	1	2	5	4	2	1	1	1	4	5	5	4	4	5	5	4	4	1	1	2	2	5	5	4
1	4	4	1	1	4	4	1		1	1	2	4	2	1	1	2	2	5	5	3	3	2	2	3	5	2	2	1	4	4	2	2	1
3	2	5	3	2	4	4	3		2	3	3	4	1	5	5	5	3	3	5	3	3	2	2	3	3	3	2	4	2	4	5	4	2
4	3	2	4	3	3	4	4		4	3	3	2	4	4	2	2	3	3	4	4	4	4	3	3	4	4	2	4	4	3	4	4	2
2	4	4	4	4	4	4	3		4	4	2	4	4	2	2	3	4	3	5	4	4	4	5	4	4	4	3	2	3	2	4	4	3
3	1	4	2	3	2	2	3		3	2	3	2	2	1	1	2	3	5	5	4	5	4	5	5	2	4	2	2	3	3	4	4	3
1	1	4	2	3	5	4	1		3	5	5	5	4	1	1	5	3	5	5	5	3	3	4	4	4	4	4	3	3	4	5	4	3
1	4	4	5	5	5	5	4		3	4	1	5	1	1	1	1	5	5	5	5	4	5	4	5	3	3	3	4	2	1	5	5	3
1	1	2	2	3	4	4	2		3	3	3	5	3	1	1	3	5	5	5	5	5	5	5	5	4	4	4	2	2	1	5	5	3
4	3	3	4	4	5	5	5		5	5	5	4	1	1	1	5	4	5	5	5	5	5	5	5	3	3	3	2	4	3	4	4	4
2	1	4	3	4	4	4	2		4	1	2	4	3	1	1	1	2	4	4	3	4	5	5	5	2	1	2	2	4	3	4	3	2
5	5	5	5	5	5	5	5		5	5	3	5	4	1	1	5	4	5	5	5	5	5	4	5	2	4	2	3	2	1	5	5	5
1	2	4	2	3	4	4	1		1	4	4	2	4	2	2	4	2	4	4	4	5	5	4	5	2	4	3	1	4	1	4	4	1
4	3	3	4	4	4	4	4		4	4	4	4	3	2	2	3	4	4	4	4	4	4	5	5	4	5	4	4	4	4	5	4	4
1	1	5	1	3	2	4	1		2	2	3	3	2	1	1	1	1	1	1	1	1	1	4	3	2	3	4	2	5	5	3	1	1
2	5	5	5	2	5	5	4		3	3	4	5	2	1	1	4	5	5	5	4	5	2	1	5	5	5	5	4	3	5	5	5	4
4	2	4	4	5	4	4	4		4	5	4	2	2	2	2	1	4	4	4	4	4	4	4	5	4	4	2	2	3	2	5	4	4
2	3	5	4	3	5	5	3		3	2	3	5	2	5	1	5	3	4	4	4	4	3	5	5	5	5	3	3	2	3	4	4	3

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
Min	1	1	1	2	1	1	2	1	1		1	1	2	1	2	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	2	1	1
1st Qu	1.5	2	3	2	3	4	4	4	2.5		2	2	2	4	2	1	1	2	2	3.5	4	4	3	4	3.5	2	3	2	2	2	4	3.5	2
Median	2	3	4	3	4	4	4	4	3		3	4	4	2	2	1	1	3	3	4	4	4	4	4	5	4	4	3	2	3	3	5	4	3
Mean	2.593	2.889	3.889	3.222	3.63	4.074	4.074	3.074		2.926	3.259	2.889	4.074	2.741	1.741	1.593	3	3.185	4.037	4.037	4	3.667	4.111	4.259	3.333	3.778	3	2.593	3.074	2.926	4.37	3.926	2.852
3rd Qu	4	4	5	4	4.5	5	4.5	4		4	4	3	5	4	4	2	2	4	4	5	4	5	4	5	5	4	4	4	4	4	5	5	5	4
Max	5	5	5	5	5	5	5	5		5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
SD	1.278799	1.476309	1.050031	1.310705	1.181529	0.957799	0.7808243	1.238048		1.268734	1.318391	1.050031	0.957799	1.227649	1.195909	1.009922	1.38675	1.075498	0.9798541	0.8077362	0.9198662	1.037749	1.002739	0.9443187	1.074172	0.9336996	1.037749	1.185141	0.957799	1.174273	0.7916948	0.9971469	1.166972
Responses	27	27	27	27	27	27	27	27	0	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27	27

Non-Robot Pretest Survey Results

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
																	1	4	4	4	3	4	4	2	4	2	2	4	3	2	2	
																	4	5	5	3	4	5	5	4	3	2		4	2	4	3	
																	4	4	4	4	4	4	4	5	4	2	2	4	2	4	4	
																	2	4	4	3	4	4	3	3	2	2	2	5	3	4	3	
																	2	3	2	2	2	3	5	5	5	4	2	4	4	4	4	
																	3	4	4	4	2	5	5	5	5	3	1	4	4	4	4	
																	4	2	4	4	4	3	3	4	5	2	1	3	2	4	3	
																	3	5	5	5	2	5	5	2	4	2	2	3	1	5	4	
																	3	4	5	4	5	4	3	4	5	2	3	3	1	5	5	
																	4	5	4	5	4	5	5	5	4	4	1	4	2	1	3	
																	2	4	4	4	4	4	4	2	3	2	4	4	2	4	4	
																	3	3	3	3	3	4	4	4	4	4	3	4	3	4	4	
																	2	3	5	4	4	5	5	5	5	3	3	4	2	5	4	
																	3	4	2	4	4	4	5	4	5	2	2	3	2	5	3	

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
Min																		1	2	2	2	2	3	3	2	2	1	3	1	1	2	
1st Qu																		2	3.25	4	3.25	3	4	3.25	3.25	4	2	3.25	2	4	3	
Median																		3	4	4	4	4	4	4	4	4	2	4	2	4	4	
Mean																		2.857	3.857	3.929	3.786	3.5	4.286	4.071	3.857	4.143	2.571	2.154	3.786	2.357	3.929	3.571	
3rd Qu																		3.75	4	4.75	4	4	5	5	5	5	3	3	4	3	4.75	4	
Max																		4	5	5	5	5	5	5	5	5	4	4	5	4	5	5	
SD																		0.9492623	0.8644378	0.997249	0.8017837	0.9405399	0.726273	0.8287419	1.167321	0.9492623	0.8516306	0.898717	0.5789342	0.9287827	1.141139	0.7559289	
Responses	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14	14	14	14	14	14	14	14	14	14	13	14	14	14	14	0

Non-Robot Posttest Survey Results

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
2	5	5	2	4	2	3	4	4										5	5	5	5	5	5	1	2	3	2	1	2	4	3	
3	5	5	4	5	4	4	3	1										5	4	3	4	4	5	4	4	4	2	2	2	5	4	
2	4	5	4	5	3	2	4	3										4	4	4	4	5	3	4	2	5	2	3	2	1	5	5	
1	4	2	1	1	3	4	1	1										1	2	2	2	2	4	4	4	5	3	2	5	4	4	4	
4	4	5	4	4	2	4	4	4										4	4	5	4	5	4	4	3	4	2	2	3	1	5	4	
1	4	4	1	1	1	4	1	1										2	5	5	5	5	5	5	2	4	3	1	4	2	2	3	
5	5	5	5	5	5	4	5	4										4	4	5	5	5	5	5	5	5	3	4	4	3	5	5	
3	2	2	3	2	4	4	3	3										3	4	4	4	4	4	4	4	3	2	3	3	4	4		
3	2	5	5	5	3	4	4	5																								

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32	Q33
Min	1	2	2	1	1	1	1	2	1	1								1	2	2	2	2	3	4	1	2	2	1	1	1	2	3	
1st Qu	2	4	4	2	2	2	2	4	3	1								2.75	4	3.75	4	4	4	4	2	4	2.75	2	2	1.75	4	3.75	
Median	3	4	5	4	4	3	3	4	4	3								4	4	4.5	4	5	4.5	4	3.5	4	3	2	3	2	4.5	4	
Mean	2.667	3.889	4.222	3.222	3.556	3	3.667	3.222	2.889									3.5	4	4.125	4.125	4.375	4.375	4.375	3.125	4.125	2.875	2.25	3	2.25	4.25	4	
3rd Qu	3	5	5	4	5	4	4	4	4									4.25	4.25	5	5	5	5	5	4	5	3	2.25	4	3	5	4.25	
Max	5	5	5	5	5	5	4	5	5									5	5	5	5	5	5	5	5	5	4	4	5	4	5	5	
SD	1.322876	1.166667	1.301708	1.563472	1.740351	1.224745	0.7071068	1.394433	1.536591									1.414214	0.9258201	1.125992	0.9910312	1.06066	0.7440238	0.5175492	1.356203	0.9910312	0.6408699	0.8864053	1.309307	1.035098	1.035098	0.7559289	
Reponses	9	9	9	9	9	9	9	9	9	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	0

IPRE Documents

Robots Course Syllabus

CS102 / ECE206 An Introduction to Computer Science

Credits and Contact Hours: 4 credits, 2.5 lecture hrs. / week, 3 lab hours / week

Instructor's or Course Coordinator's Name: Bruce MacLennan

Textbook and Other Supplemental Material

a. *Learning Computing with Robots in C++*, ed. by Deepak Kumar, 2010.
b. *How to Think Like a Computer Scientist: Learning with C++*, by Allen B. Downey, 2010.
c. online notes and example programs.

Specific Course Information

a. **Catalog Description**: Problem solving and algorithm development. Organization and characteristics of modern digital computers with emphasis on software engineering, building abstractions with procedures and data, and programming in a modern computer language. Includes Level 1 design projects, which require laboratory work.
b. **Prerequisites** and/or **Co-requisites:** none
c. **Required**

Specific Goals for the Course

a. specific outcomes of instruction:
(1) Students will gain experience in algorithmic problem solving.
(2) Students will be able to understand and apply basic programming techniques.
(3) Students will understand and apply basic approaches to program development.
(4) Students will be able to read and write basic C++ programs.

b. Student Outcomes listed in Criterion 3

	Engineering Accreditation Commission Statement of Student Outcomes												
Course Outcomes	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l#1**	**l#2**
alg. problem solving			**X**		**X**								
basic prog. techniques			**X**								**X**		
program development			**X**								**X**		
C++ prog.											**X**		

or:

	Computing Accreditation Commission Statement of Student Outcomes										
Course Outcomes	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**
alg. problem solving		**X**	**X**							**X**	
basic prog. techniques			**X**						**X**		**X**
program development			**X**						**X**		**X**
C++ prog.									**X**		

Topics Covered

basic programming concepts (arithmetic, boolean logic, assignment, conditional execution, iteration, input-output, function definition, recursion, pseudo-random numbers, pointers, dynamic memory management, linked lists), **C++ syntax and semantics** (numbers, characters, strings, expressions, function definition, if-else, switch, for, while, <<, >>, file I/O, formatted I/O, strings, vectors, structs, enumerations, typedef, pass by value, reference, and constant reference; STL: vectors, lists), **object-oriented programming** (classes, member functions, constructors, subclasses and inheritance, public, private, protected, virtual definition, overloading, templates), **program development** (top-down, bottom-up, incremental, modular design, unit testing, input validity checking, preconditions, invariants and assertions), **sorting and searching** (bubble, selection, mergesort, sequential search, bisection search), **robotics and AI** (Braitenberg vehicles, reactive control, behavior-based control, game playing, look ahead).

Proposed Study Documents

Documents Submitted for IRB Approval

Recruitment Poster

Consent Form

Consent Form

Researcher: Shannon Gray
Faculty Adviser: Keith O'Hara

I am a student at Bard College, and I am conducting programming workshops as an experiment for my Senior Project. I am studying the amount of Computer Science that can be taught in an hour, in association with The Hour of Code campaign from Code.org during Computer Science Education Week (December 8-14).

During this workshop, you will complete two programming tasks. You will also be asked to complete pretest and posttest surveys to gauge your Computer Science background and learning gains from the workshop respectively. This entire process was designed to be approximately an hour and a half in length.

All information will be kept confidential. I will keep the data in a secure place. Only myself and the faculty supervisor mentioned above will have access to this information. Upon completion of this project, all data will be destroyed or stored in a secure location.

There are minor foreseeable risks involved in this workshop. You may experience minor frustration or eye strain during this workshop from working on the computers. Benefits of this workshop include learning to code, contributing to scientific research, and helping a fellow Bard student complete their Senior Project.

By participating in this workshop, you will be entered into a raffle to receive one of two $50 Amazon Giftcards. If you decide to leave the workshop at any time, you will still be entered into the raffle, regardless of completion of the workshop.

Participant's Agreement:

I am aware that my participation in this workshop is voluntary. I understand the intent and purpose of this research. If, for any reason, at any time, I wish to stop participating in the workshop, I may do so without having to give an explanation.

The researcher has reviewed the individual and social benefits and risks of this project with me.

I am aware the data will be used in a Senior Project that will be publicly available at the Stevenson Library on the Bard College Campus. I have the right to review, comment on,

and/or withdraw information prior to the Senior Project's submission. The data gathered in this study are confidential with respect to my personal identity unless I specify otherwise.

If I have any questions about this study, I am free to contact the student researcher (Shannon Gray, sg6337@bard.edu, 845-853-6866) or the faculty adviser (Prof. Keith O'Hara, kohara@bard.edu, 845-752-2359). If I have any questions about my rights as a research participant, I am free to contact the chair of Institutional Review Board: Michelle Murray (IRB@bard.edu; mkmurray@bard.edu, 845-758-7693).

I have been offered a copy of this consent form that I may keep for my own reference.

I have read the above form and, with the understanding that I can withdraw at any time and for whatever reason, I consent to participate in today's workshop.

_____ _____

Participant's signature Date

_____

Interviewer's signature

Pretest Survey

Participant Survey

Thank you for participating in this study. Your individual responses will be kept confidential and WILL NOT affect your chances in the raffle. Please fill in each oval completely and be as honest and accurate as possible.

Demographic Information

1. What is your major?
O _____

2. What is your gender?
O _____

3. What year are you?
O 1st (undergraduate)
O 2nd (undergraduate)
O 3rd (undergraduate)
O 4th or more (undergraduate)
O Graduate

4. Have you written a computer program (of any size) before?
O Yes O No

5. Which programming languages have you used before this class (fill in all that apply)?
O None O Scheme O Visual Basic
O Java O Python O C/C++
O Other: _____

6. Rate your prior overall programming experience.
O No experience O Beginner
O Intermediate O Advanced

7. What type of computer / OS are you using?
O _____

Attitudes: Please use the scale below to rate the following statements.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
8. Compared to students in this workshop, I feel I know a lot about computers.	O	O	O	O	O
9. I am confident in my problem solving ability.	O	O	O	O	O
10. I am confident in my ability to persist until a solution is found.	O	O	O	O	O
11. I am confident in my ability to meet unexpected challenges with success.	O	O	O	O	O
12. I enjoy being challenged by seemingly unsolvable situations or problems.	O	O	O	O	O

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
13. I am confident in my math ability.	O	O	O	O	O
14. I am confident in my science reasoning ability.	O	O	O	O	O
15. I seek help in class before I become very frustrated.	O	O	O	O	O
16. I am not afraid to ask for help.	O	O	O	O	O
17. Computer Science and programming are the same thing.	O	O	O	O	O
18. I took this workshop to see what computer science is all about.	O	O	O	O	O
19. Other students in this workshop seem to know more about computers than I do.	O	O	O	O	O
20. I dislike situations or problems that are seemingly unsolvable.	O	O	O	O	O
21. I like technology.	O	O	O	O	O
22. I regularly try out new experiences and products.	O	O	O	O	O

Robots Posttest Survey

Participant Survey

Thank you for participating in this study. Your individual responses will be kept confidential and WILL NOT affect your chances in the raffle. Please fill in each oval completely and be as honest and accurate as possible.

Demographic Information

1. What is your major?
O _____

2. What is your gender?
O _____

3. What year are you?
O 1st (undergraduate)
O 2nd (undergraduate)
O 3rd (undergraduate)
O 4th or more (undergraduate)
O Graduate

Workshop: Please use the scale below to rate the following statements.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
4. My experiences in this workshop have caused me to decide to take a computer science class.	O	O	O	O	O
5. My perceptions of computer science have changed negatively after taking this workshop.	O	O	O	O	O
6. My perceptions of computer science have changed positively after taking this workshop.	O	O	O	O	O
7. I plan to find out more about computer science in the future.	O	O	O	O	O
8. What I learned in this workshop is important to my future career.	O	O	O	O	O
9. I discussed parts of the workshop with friends also in the workshop.	O	O	O	O	O
10. I will talk with my friends (not in the workshop) about the workshop.	O	O	O	O	O
11. I enjoyed this workshop.	O	O	O	O	O
12. I enjoyed using the robot in this workshop.	O	O	O	O	O
13. Using the robot was easy.	O	O	O	O	O
14. Using a simulated robot would have been more interesting/worthwhile than a real one.	O	O	O	O	O

Attitudes: Please use the scale below to rate the following statements.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
15. Compared to students in this workshop, I feel I know a lot about computers.	O	O	O	O	O
16. I am confident in my problem solving ability.	O	O	O	O	O
17. I am confident in my ability to persist until a solution is found.	O	O	O	O	O
18. I am confident in my ability to meet unexpected challenges with success.	O	O	O	O	O
19. I enjoy being challenged by seemingly unsolvable situations or problems.	O	O	O	O	O
20. I am confident in my math ability.	O	O	O	O	O
21. I am confident in my science reasoning ability.	O	O	O	O	O
22. I seek help in class before I become very frustrated.	O	O	O	O	O
23. I am not afraid to ask for help.	O	O	O	O	O
24. Computer Science and programming are the same thing.	O	O	O	O	O
25. I took this workshop to see what computer science is all about.	O	O	O	O	O
26. Other students in this workshop seem to know more about computers than I do.	O	O	O	O	O
27. I dislike situations or problems that are seemingly unsolvable.	O	O	O	O	O
28. I like technology.	O	O	O	O	O
29. I regularly try out new experiences and products.	O	O	O	O	O

Simulation Posttest Survey

Participant Survey

Thank you for participating in this study. Your individual responses will be kept confidential and WILL NOT affect your chances in the raffle. Please fill in each oval completely and be as honest and accurate as possible.

Demographic Information

1. What is your major?
O _____

2. What is your gender?
O _____

3. What year are you?
O 1st (undergraduate)
O 2nd (undergraduate)
O 3rd (undergraduate)
O 4th or more (undergraduate)
O Graduate

Workshop: Please use the scale below to rate the following statements.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
4. My experiences in this workshop have caused me to decide to take a computer science class.	O	O	O	O	O
5. My perceptions of computer science have changed negatively after taking this workshop.	O	O	O	O	O
6. My perceptions of computer science have changed positively after taking this workshop.	O	O	O	O	O
7. I plan to find out more about computer science in the future.	O	O	O	O	O
8. What I learned in this workshop is important to my future career.	O	O	O	O	O
9. I discussed parts of the workshop with friends also in the workshop.	O	O	O	O	O
10. I will talk with my friends (not in the workshop) about the workshop.	O	O	O	O	O
11. I enjoyed this workshop.	O	O	O	O	O
12. I enjoyed using the simulated robot in this workshop.	O	O	O	O	O
13. Using the simulated robot was easy.	O	O	O	O	O
14. Using a real robot would have been more interesting/worthwhile than a simulated one.	O	O	O	O	O

Attitudes: Please use the scale below to rate the following statements.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
15. Compared to students in this workshop, I feel I know a lot about computers.	O	O	O	O	O
16. I am confident in my problem solving ability.	O	O	O	O	O
17. I am confident in my ability to persist until a solution is found.	O	O	O	O	O
18. I am confident in my ability to meet unexpected challenges with success.	O	O	O	O	O
19. I enjoy being challenged by seemingly unsolvable situations or problems.	O	O	O	O	O
20. I am confident in my math ability.	O	O	O	O	O
21. I am confident in my science reasoning ability.	O	O	O	O	O
22. I seek help in class before I become very frustrated.	O	O	O	O	O
23. I am not afraid to ask for help.	O	O	O	O	O
24. Computer Science and programming are the same thing.	O	O	O	O	O
25. I took this workshop to see what computer science is all about.	O	O	O	O	O
26. Other students in this workshop seem to know more about computers than I do.	O	O	O	O	O
27. I dislike situations or problems that are seemingly unsolvable.	O	O	O	O	O
28. I like technology.	O	O	O	O	O
29. I regularly try out new experiences and products.	O	O	O	O	O

Contact Form

Contact Information

If you have any questions about this study or your participation in this study, feel free to contact any of the following individuals.

Shannon Gray
Researcher
sg6337@bard.edu
(845) 853-6866

Keith O'Hara
Faculty Advisor
kohara@bard.edu
(845) 752-2359

Michelle Murray
Chair, Bard Institutional Review Board
mkmurray@bard.edu
(845) 758-7693

The Bard Institutional Review Board may also be reached at IRB@bard.edu.

Debriefing Form

Debriefing Form

This study was looking into how much computer science a student could actually learn in an hour, but I was also interested in a few other ideas as well. I wanted to know if an hour of computer science was enough to change people's perceptions of the subject. This is why I had you do the pre and post surveys. I was also interested in the differences between using real robots to teach computer science versus simulated ones. I wanted to see if one generated more interest than the other, or if one is a better teaching tool to increase students' learning and/or enjoyment. This is why some workshops used real robots and some used simulated ones.

Thanks again for helping with my Senior Project by participating in these workshops!

Workshop Introduction Materials

Workshop Script

Script

{FIRST SLIDE UP}

Hello and welcome to my Calico Coding Workshops. Thank you so much for participating. We will start off this workshop by going over the consent process, and filling out a short Attitudes Survey.

[hand out both forms - precoded]

Your participation in this workshop is expected to take about an hour and a half. During this time you'll complete two programming tasks. If you're ever uncomfortable for any reason and would like to stop participating, that is OK, just say so. You will still be entered into the raffle, regardless of the completion of the workshop. Your data will be stored according to a coding number, so your responses will remain confidential. Before we start you need to read this consent form carefully. Consent forms are necessary so that you are accurately informed about the research process and you understand your rights. If you have any questions, just ask me. Then, if you agree with the content of the consent form, sign at the bottom. If you'd like a copy of the consent form, you may take a copy with you.

[collect consent forms]

Now will you please fill out the Participant Survey as well.

[collect pretests]

Now that we've gotten that out of the way, we can begin our basic introduction to Calico and Python.

{NEXT SLIDE}

Calico

Calico allows you to control the Scribbler robots, which we will be using in this workshop. It can do a medley of other cool things as well, but we will be focusing on this aspect of Calico. Here is what the Calico interface looks like.

[Open and Show Calico]

At the top left of the screen is the shell. This area is used to input commands, which I will demonstrate later on. Below that, on the lower left side is the Output window. This will print out any output your program may have. I will also demonstrate this. On the right side is where you can actually write code. You can load files or start new ones to work on.

To run programs in Calico you press this green button up here.

[Indicate Green Button]

As you can see, when I pressed that button, my program began to run, and it printed output in the output window. But make sure you saved any recent changes you made, or Calico will run the code without them. You can save using this floppy-disk button here.

[indicate save button]

Calico also has the useful feature of being able to visualize a program as it is running. You can accomplish this using the slider bar, that is located next to the run button.

[Indicate slider bar, move it around]

This bar controls how fast the visualizations will occur. Sliding to the right will make it go slower and sliding to the left will make it go faster. I'm going to give it a relatively slow speed to that we can see how the program is running.

[Set slider to the right and run program again.]

Now you can see that the program is highlighting the line of code it is currently running. Those are the basics of Calico.

[Open example 1]

Here is an example python program. One thing to notice are the different levels of tabs here. Python uses indents or tabs to organize code. The whitespace indentation affects how the code is run. A logical block of statements, such as the ones I have here, should all have indentation according to their "parent" statement. For instance, this if statement is tabbed in because it "child" statement, and belongs under this for loop, the "parent" statement. Likewise, this print statement is tabbed over because it is a "child" statement of the if statement. And this last statement, the x=x+1 is tabbed back because it does not belong inside the if statement, but it does belong inside the for loop. It is another "child" statement of the for loop.

If one of the lines in a group has a different indentation, it is marked as an error.

{NEXT SLIDE}

Basic Commands

I will now go over some of the basic commands you will be using in this workshop.

[Open example 2]

The first two we will look at are motors() and wait(). The motors() command takes values for the left wheel speed and right wheel speed. These values can be anywhere from 1 to -1. Generally 1 means go forward and -1 means go backward. The wait() command makes the previous command continue to happen for a certain amount of time. So for instance, I have motors(1,1) which will make the robot go forward at a speed of one, and then that is followed by wait(3). All together this will have the robot drive forward for 3 seconds. Motors(0,0) will then set both wheel speeds to 0, and stop the robot from moving.

[run example 2]

You can also use the shell on the top left side of the screen to type in these commands. If I write motors(1,1) and press enter, the robot will move forward with left wheel speed of 1 and right wheel speed of 1.

[Do this]

And now it has run into the wall. I can type in motors(-1,-1), and the robot will go the opposite direction. And if I want the robot to stop I will type in motors(0,0) and the robot will stop.

[Do this too]

{NEXT SLIDE}

The next topic we will be discussing is Loops. Loops make something happen multiple times in a row. For instance the loop I have here, for seconds in timer(60), will have the code written below it run for 60 seconds. The code I have here is a basic wall-avoiding program. The robot drives around and if it detects a wall using its IR sensors it will stop and move in a different direction. Here is an example of this program running.

[run example 3]

The robot will drive around and avoid walls for 60 seconds because that is what we set in the loop. But we get the point of that, so we don't have to watch the robot drive around anymore. Notice that I could just as easily put in a different inequality value here. The code wouldn't behave the same way as we just saw, but for example, you could change this EqualsEquals to a LessThan, or a GreatThanOrEqual to sign instead. You can modify this to have whatever relationship you want.

[show example, put in lessThan or greaterThanorEqual]

{NEXT SLIDE}

The next topic we will be discussing is Conditionals. Conditionals make things happen if a

specified condition is met. That means if the condition specified in the if statement is met, the code below will run, and if it is not met, the code will not run. For example, if condition A is met then condition B will occur.

Looking again at the wall-avoid code, you can see I have two conditional if statements. The if getIR(Left) and the if getIR(right). These control which direction my robot turns. If he senses an object with his left IR sensor, then he will turn to the right. Likewise, if he senses an object to his right, he will turn left. The turning occurs only when these conditions are met.

We will now move onto the goals of the workshop.

{NEXT SLIDE}

The first goal of this workshop is to create a dance sequence that lasts for *at least 20 seconds*. This can be as simple or as complicated as you like. You can use loops, you can make the robot go in circles, zigzags, whatever you want . This is mainly to get you warmed up with using Calico and Python.

{NEXT SLIDE}

The next task is to create a light finder program. For this you will have the robot try to find a light source in the environment, and beep when it comes "into view" of the robot, or just close enough to it. This is so we can know that the robot has actually found the light. Both of these tasks will be explained in more detail on the handout which you will receive.

{NEXT SLIDE}

Remember, you can always ask questions if you get stuck or are just curious about something. You can also go to either of these websites to get more information about Python as well. I will leave this screen up, but you can find these two links on your cheat sheet as well.

Good luck, and thanks again for participating!

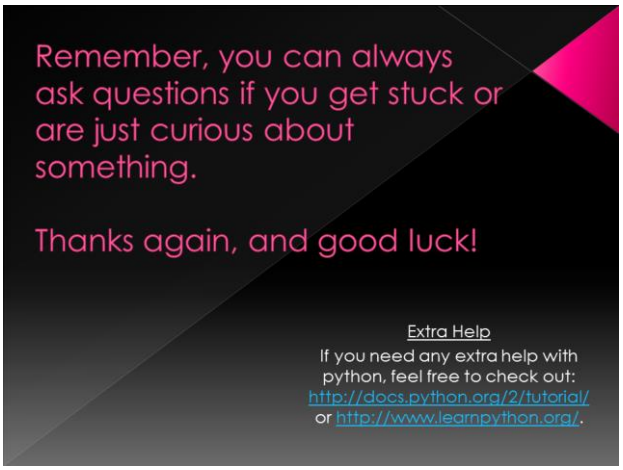
Schedule:
0-5: Welcome and consent forms
5-15: Pretest surveys
15-25: Intro
25-45: Dance
45 - 1:15: Light Follow
1:15 - 1:30: Posttest surveys

0 - 30: Set up next workshop
Rinse and Repeat.

PowerPoint Slides


Goals for this Workshop
1) Dance:
You will create a simple (or not so simple) dance sequence for your robot that lasts at least twenty seconds.
This should get you warmed up with using Calico and Python.


Goals for this Workshop
Light Finder:
You will create a program that has the robot try to find a light source, and beep once it gets close to it.


Remember, you can always ask questions if you get stuck or are just curious about something.
Thanks again, and good luck!
Extra Help
If you need any extra help with python, feel free to check out:
http://docs.python.org/2/tutorial/
or http://www.learnpython.org/.

Workshop Task Materials

Workshop "Cheat Sheet"

Calico "Cheat" Sheet

General

- You can run Calico programs by pressing the green button at the top of the screen.
- You can use the slider bar next to the green run button to control how fast the visualization of the program occurs.
- *Extra Help:*
  - If you need any extra help with python, feel free to check out: http://docs.python.org/2/tutorial/ or http://www.learnpython.org/.

Commands

- **motors(left, right):** Left and right are the speeds of the left and right wheels. These values can be anywhere from 1 to -1 inclusively.
  - *Example: motors(1,1) will have the robot go forward with both wheel speeds set to one. motors(-1,-1) will have the robot go backwards with both wheel speeds set to negative one.*
- **wait(duration):** This has the previous command continue for a set amount of time in seconds.
  - *Example: motors(1,1)*
    wait(2)
    will have the motors(1,1) command continue to run for two seconds.
- **getIR("left"), getIR("right"):** Gets the IR value of the left or right sensor. If an object is in front of the sensor the value will be zero, otherwise it will be one.
- **getLight("left"), getLight("center"), getLight("right"):** Gets the light values of the three sensors. Values will be lower when a light source is in view, and higher when a light source is not in view.
- **beep(duration, frequency):** Has the robot beep for a certain amount of seconds at a set frequency.
  - *Example: beep(2, 500)*

Concepts

- **Loops - for seconds in timer(duration):** This line of code will make something happen over and over again for however many seconds are given for a duration.
  - *Example: for seconds in timer(60):*
    print("Hi.")
    prints out the message "Hi." repeatedly for sixty seconds and then stop.
- **Conditonals - if statements:** These allow you to have code execute conditionally. That means if the condition specified in the if statement is met, the code below will run, and if it is not met, the code will not run. For example, if condition A is met then condition B will occur.
  - *Example: if x > 10:*
    print(x)
    will print out x as long as it is less than the value ten.

Goals for this workshop

- **Goal 1, Dance:** You will create a simple (or not so simple, it's up to you!) dance sequence for your robot that lasts *at least twenty seconds*. You can have your robot move in a square or a circle, do loops or zigzags, or whatever else you wish! This should get you warmed up with using Calico and Python.

- **Goal 2, Light Finder:** For this task you will create a light finder program, where the robot drives around trying to find a light source. When the light comes into the view of the robot, you will have the robot beep, so that you can tell the robot has seen the light.

Remember, you can always ask questions if you get stuck or are just curious about something.
Thanks again, and good luck!

BIBLIOGRAPHY

[1] "ABC's Influence on Python." *The Making of Python*. N.p., n.d. Web. 17 Apr. 2014. <http://www.artima.com/intv/pythonP.html>.

[2] Blank, D., J. S. Kay, J. B. Marshall, K. O'Hara, and M. Russo. "Calico: A Multi-Programming-Language, Multi-Context Framework Designed for Computer Science Education." *SIGCSE '12 Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (n.d.): 63-68. Web. 16 Sept. 2013.

[3] Cheryan, S., V. C. Plaut, C. Handron, and L. Hudson. "The Stereotypical Computer Scientist: Gendered Media Representations as a Barrier to Inclusion for Women." *Sex Roles: A Journal of Research* 69.1-2 (2013): 58-71. Web. 7 Mar. 2014.

[4] "Computer Science...without a Computer!" *Computer Science Unplugged*. N.p., n.d. Web. 25 Apr. 2014. <http://csunplugged.org/>.

[5] Correll, N., C. Wailes, and S. Slaby. "A One-hour Curriculum to Engage Middle School Students in Robotics and Computer Science Using Cubelets." (2012): 1-12. Web. 23 Sept. 2013.

[6] "ECE 206 Introduction to Computer Science." *ECE 206 Introduction to Computer Science*. N.p., n.d. Web. 20 Apr. 2014. <http://web.eecs.utk.edu/~cs102/>.

[7] uzdial, M. "Is It worth Teaching Young Kids to Code?" Web log post. *Computing Education Blog*. N.p., n.d. Web. 7 Oct. 2013.

[8] "The Hour of Code." *Computer Science Education Week*. N.p., 2014. Web. 7 Mar. 2014. <http://csedweek.org/>.

[9] "Institute for Personal Robots in Education 2007 Annual Report." (2007): n. pag. Web. 18 Apr. 2014.

[10] "Institute for Personal Robots in Education." *Institute for Personal Robots in Education*. N.p., n.d. Web. 2 Jan. 2014. <http://www.roboteducation.org/>.

[11] "An Introduction to Computer Scienceusing Robots!" *CS102 / ECE 206 — Introduction to Computer Science Using Robots*. N.p., n.d. Web. 20 Apr. 2014. <http://web.eecs.utk.edu/~mclennan/Classes/102/>.

[12] Kumar, D., D. Blank, T. Balch, K. O'Hara, M. Guzdial, and S. Tansley. "Engaging Computing Students with AI and Robotics." *Association for the Advancement of Artificial Intelligence Spring Symposia* (2008): n. pag. Web. 13 Apr. 2014.

[13] Lee, J., and A. Howard. "Robotic Adventure Games to Enhance Understanding of Computer Science for Middle School Students." (2008): 1-6. Web. 18 Oct. 2013.

[14] "Main Page." *IPRE Wiki*. N.p., n.d. Web. 17 Feb. 2014. <http://wiki.roboteducation.org/Main_Page>.

[15] Maloney, J., M. Resnick, N. Rusk, B. SIlverman, and E. Eastmond. "The Scratch Programming Language and Environment." *ACM Transactions on Computing Education (TOCE)* Article 16 10.4 (2010): 1-15. Web. 16 Sept. 2013.

[16] Martin, F. G. "Real Robots Don't Drive Straight." *American Association for Artificial Intelligence Spring Symposium* (2007): 1-5. Web. 18 Oct. 2013.

[17] Mazur, N., and M. Kuhrt. "Teaching Programming Concepts Using a Robot Simulation." *CCSC: Northeastern Conference* (1997): 4-11. Web. 6 Oct. 2013.

[18] McGill, M. M. "Learning to Program with Personal Robots: Influences on Student Motivation." *ACM Transactions on Computing Education (TOCE)* 4th ser. 12.1 (2012): n. pag. Web. 2 Apr. 2014.

[19] "Parallax's Red Hot Scribbler 2 Robot." *Robot Magazine*. N.p., n.d. Web. 9 Mar. 2014. <http://www.botmag.com/index.php/parallaxs-red-hot-scribbler-2-robot>.

[20] Prensky, M. ""Simulations": Are They Games?" (2001): 1-10. Web. 18 Oct. 2013.

[21] "Review: Scribbler Robot." *Robot News and Robotics Info*. N.p., n.d. Web. 3 Jan. 2014. <http://robots.net/article/1729.html>.

[22] Sanner, M. F. "Python: A Programming Language for Software." *Journal of Molecular Graphics & Modelling* 17.1 (1999): 57-61. Web. 25 Apr. 2014.

[23] Summet, J., D. Kumar, K. O'hara, D. Walker, L. Ni, D. Blank, and T. Balch. "Personalizing CS1 with Robots." *ACM SIGCSE Bulletin* 41.1 (2009): 433-37. Web. 2 Apr. 2014.

[24] "Teach the Hour of Code™ in Your Classroom." *Code.org*. N.p., 2014. Web. 7 Mar. 2014. <https://code.org/educate/hoc>.

[25] Tucker, A., F. Deek, J. Jones, D. McCowan, C. Stephanson, and A. Verno. "A Model Curriculum for K–12 Computer Science." *Final Report of the ACM K–12 Task Force Curriculum Committee* (2003): 1-45. Web. 23 Sept. 2013.

[26] Wyeth, P., and H. C. Purchase. "Programming Without a Computer: A New Interface for Children under Eight." *AUIC '00 Proceedings of the First Australasian User Interface Conference* (2000): 1-8. Web. 23 Sept. 2013.