INTEGRATED COMPUTER SCIENCE UNIT FOR EARLY EXPOSURE TO A DIVERSE POPULATION

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Di Hoang

June 2018

COMMITTEE MEMBERSHIP

TITLE:	Integrated Computer Science Unit for Early Exposure to A Diverse Population
AUTHOR:	Di Hoang
DATE SUBMITTED:	June 2018
COMMITTEE CHAIR:	Aaron Keen, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Theresa Anne Migler-VonDollen, Ph.D. Lecturer of Computer Science
COMMITTEE MEMBER:	Maria Pantoja, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Ignatios Vakalis, Ph.D. Professor of Computer Science

ABSTRACT

Integrated Computer Science Unit for Early Exposure to A Diverse Population

Di Hoang

Reports from major tech companies, such as Airbnb, Facebook, Microsoft, Google, Apple, and Yahoo, indicate that gender diversity and ethnic diversity are lacking in the tech industry [22]. The US Bureau of Labor Statistics projected a 24% increase in the number of software development positions (roughly an increase of 300,000 jobs) between the years 2016 and 2026 [64]. The main motivation behind this thesis is to expose computer science to young children and encourage teachers to include computer science in their classrooms in the hope of contributing to the current effort of increasing diversity within tech fields in the next decade.

This thesis presents an Integrated Computer Science Unit and a teaching tool. Lessons in the Integrated Computer Science Unit utilize computer science as a tool to interactively demonstrate the lesson materials. The teaching tool is designed to help teachers generate computer science integrated lessons and assess students' work.

ACKNOWLEDGMENTS

Thanks to:

- David Morando, for being my biggest support and helping me with my thesis work in the classrooms
- Dr. Aaron Keen, for being my advisor and helping me every step of the way
- Dr. Maria Pantoja, Dr. Theresa Migler-VonDollen, and Dr. Ignatios Vakalis, for being on my committee
- Kim Bisheff and Pacheco Elementary School, for allowing me to borrow Pacheco Code Club for my thesis work
- Kathryn McDonald and Lange Elementary School, for allowing me to borrow her classroom for my thesis work
- Molly Mackay and Grover Heights Elementary School, for allowing me to borrow her classroom for my thesis work
- My friends, WISH, and the CSSE department, for their support throughout my time at Cal Poly
- My parents, for giving me the opportunities I have today
- My cats, Peanut and Walnut, for making me happy everyday
- Andrew Guenther, for uploading this template

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

LIST OF FIGURES

Figure Page

LIST OF LISTINGS

Chapter 1

INTRODUCTION

Computers and technology affect many aspects of our lives. Retail, health care, banking, airlines, social sciences, and marketing, to name a few, all operate today using different forms of software. As the field of computer science gains popularity and the job outlook for the field continues to boom, it is important for children, from all backgrounds, to learn about the things that surround their daily lives.

Since the 1970s, the field of computing has been dominated by men. The percentage of women pursuing a degree in computer science peaked in 1985 at 36.8%, but the number has dropped since then to 18% as of 2015 [17]. Even at a young age, girls are less often encouraged than boys to pursue STEM (Science, Technology, Engineering, and Mathematics) disciplines and they grow up without considering careers in those fields as an option [28, 53, 62]. Similarly, other underrepresented groups are not frequently exposed to career options in STEM. Parental occupation in a STEM field is one of the main predictors of children majoring in STEM in college [54]. A survey of 2,000 parents showed that while 93% of the parents claimed to know what engineering is, only 23% indicated that they are confident to describe what an engineer does [55]. These findings suggest that parents who work in STEM are more able to encourage their children to pursue STEM, while parents who are not in STEM generally do not know enough about the industry to promote it to their children. Focusing on the tech industry, which seeks talent from computer science and various types of engineering, the number of minorities is extremely low [35].

The movement to increase diversity in tech has been set in motion and many tech companies are participating. Many companies sponsor and recruit at conferences geared towards women, such as the Grace Hopper Conference [29], and also offer scholarships to encourage minorities to pursue STEM careers [7, 48]. A handful of companies, such as Apple, Facebook, Google, and Netflix, have announced new policies to set up breastfeeding rooms, to give expectant mothers longer paid maternity leave, and to provide monetary support for unexpected costs associated with a newborn [65]. Some companies, like Amazon and Google, have different diversity groups at the workplace to provide their employees with opportunities for mentorship and networking [4, 32]. Despite the effort, the numbers of minority groups in tech are still lacking.

According to self-reported statistics from Airbnb [3], Facebook [26], Microsoft [47], Google [31], Apple [6], and Yahoo [68], displayed in Figure 1.1 [22], 48-61% of employees are White and 25-43% are Asian. Only 2-8% are Hispanic, 1-7% are Black, and 2-5% are of other races. Figure 1.2 displays the numbers of men and women in tech gathered from Equal Employment Opportunities reports from 2015 for the same tech companies as in Figure 1.1 [2, 25, 5, 67]. The smallest difference is 8% but the majority of the differences are between 38-52%. It is clear that there is an overwhelmingly larger percentage of men than women. It is important to strive for gender diversity because there are many benefits to a gender-balanced team such as greater productivity, better performance, and higher revenues [27].

1.1 Problem

Many students, especially those from minority families, do not have a great chance to learn about computer science as they are growing up. As the time for college application approaches, such students may not be aware of the field or they may not know what careers they could pursue by obtaining a degree in Computer Science. Primary school students generally do not learn about computer science in school since the subject is not required in the curriculum. Currently there does not exist a state-

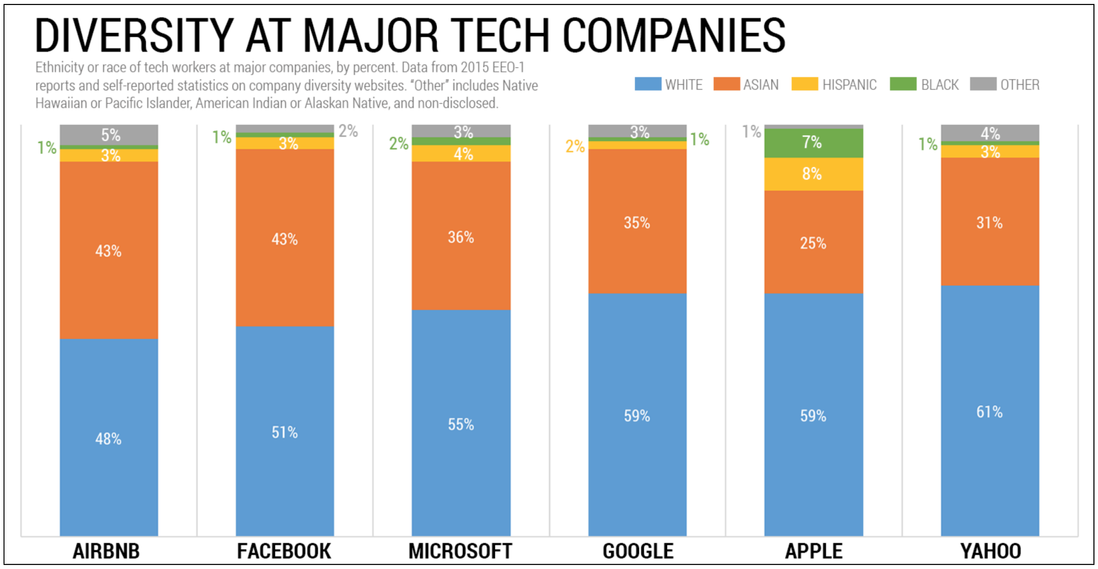


Figure 1.1: Diversity At Major Tech Companies, 2015 [22]

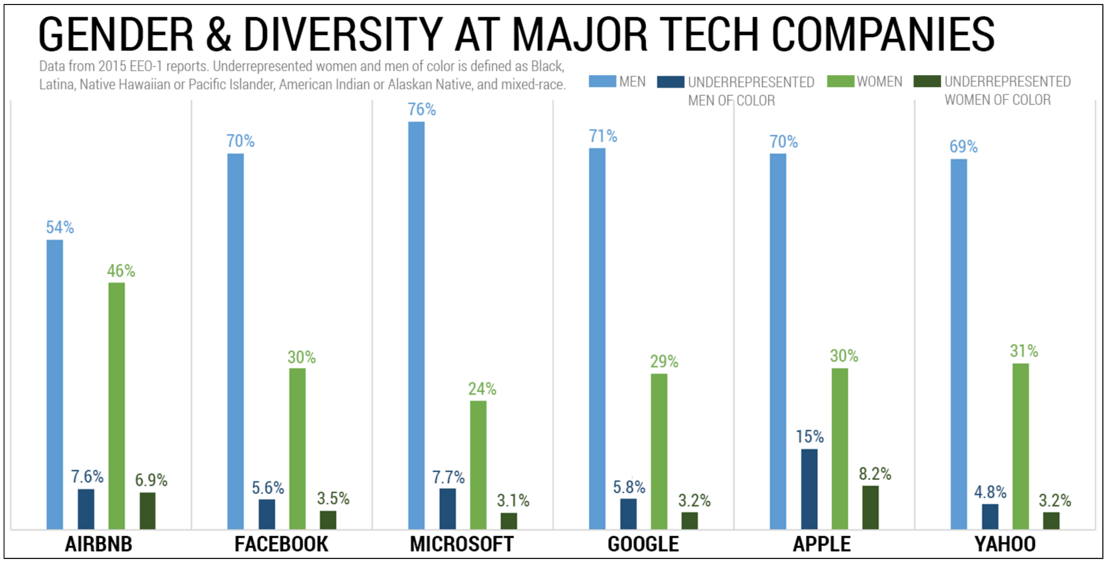


Figure 1.2: Gender At Major Tech Companies, 2015 [22]

wide computer science curriculum, although a set of standards have been drafted and is awaiting the approval process [20].

56% of teachers across the nation believe that students should be required to learn computer science [14], but time is the main stressor for many teachers [56]. One teacher from Grover Heights Elementary School started a coding program that occurred before school for an hour a week where students followed the Code.org [13] curriculum independently, but she could not spend the extra time for the coding hour any longer as her schedule became busier.

1.2 Context

Several teachers have informed me that they reserve approximately an hour each week for students to spend in the computer lab. Activities during computer lab time allow students to learn about different subjects as they get to practice working with a computer. Students learn how to use their mouse to move around the screen, how to use their keyboard to type, and other basic shortcuts like copy and paste. Teachers indicate that computer lab activities usually include making presentation slides, writing and editing on Google Docs, learning how to do research on the Internet, and playing games on Kahoot! [38]. Other students who have Chromebooks available in their classroom get more computer time, but the activities are generally similar. This designated time makes it convenient for incorporating a computer science unit.

1.3 Contribution

The work in this thesis provides various means to encourage the teaching of computer science in elementary school classrooms. The work is divided into two parts:

1. Integrated Computer Science Unit: a set of four interactive computer science activities that can be integrated with other mandatory subjects that are taught in school. The primary contribution of the computer science unit is to provide a set of lessons that highlight the importance of computer science as a tool to help solve problems in other subjects. The unit aims to teach students minimal coding with the main emphasis on the applications of computer science. This

set of lessons could help alleviate some of the time management burden for teachers by allowing them to teach the required standards and at the same time introduce their students to computer science to get them interested in careers in the field. Furthermore, all lessons in the unit are accessible, affordable, and require minimal equipment.

2. Teaching Tool: a web-based tool to help teachers generate new lessons based on two of the activities in the computer science unit.

The ultimate goal is to expose young students to programming and teach them that the field has a wide range of applications. In doing so, the students are able to see that their favorite subjects and computing can go hand-in-hand. This may help motivate students to apply their passions and consider pursuing a tech based career in the future.

Chapter 2

BACKGROUND

2.1 Standards

Standards are learning goals that outline what a student should know and be able to do at the end of each grade throughout K-12 [16]. This section discusses the different standards that are integrated into the Integrated Computer Science Unit.

2.1.1 Common Core State Standards

The Common Core State Standards were established in order to create a consistent set of standards across the nation. It provides a set of "high-quality academic standards in mathematics and English language arts (ELA)" [16]. The initiative aims to prepare students for success in college and the workplace and was adopted by most states, including California, in 2010.

The Common Core English Language Arts standards are divided into Reading, Writing, Speaking and Listening, and Language. The standards define what it means to be literate in the 21st century, meaning that students will learn to read and analyze both print and digital texts, use technology for research, and share viewpoints in collaborative conversations [23].

The Common Core Mathematics standards aim to teach students real-world mathematical applications in the 21st century. The standards call for conceptual understanding as well as the ability to apply mathematics and aim to ensure that students are "prepared for college, careers, and productive citizenship" [46].

2.1.2 California History-Social Science Standards

The California History-Social Science Standards (HSS) was created in 2000 with the motivation of redefining the state's expectations for history-social science education. The standards focus on the story of America in a chronological order but also call for analyzing relationships with other countries and regions in the world [36].

2.1.3 California Next Generation Science Standards

California adopted the Next Generation Science Standards (NGSS) in 2013 [58]. The distinction of NGSS from previous state standards is that it follows the three-dimension approach [49]. The three dimensions are Disciplinary Core Ideas, Cross-Cutting Concepts, and Practices. Core Ideas in this discipline are divided into four domains: Physical Science, Life Science, Earth and Space Science, and Engineering. Cross-Cutting concepts allow students to make connections between the four domains to obtain a full view of the world around them. Practices help students strengthen their knowledge of the core ideas and cross-cutting concepts by showing them how scientists investigate the world and how engineers build systems.

2.2 Integration

A lesson that integrates one subject with another subject is also known as a cross-curricular lesson. Cross-curricular is defined as "a conscious effort to apply knowledge, principles, and/or values to more than one academic discipline simultaneously" [18]. The idea of integrating multiple subjects is, in fact, common and many school subjects are already integrated with each other, such as Math/Science integration, Math/History-Social Science integration, Language Arts/Science integration, and Language Arts/History-Social Science integration [19]. Some examples are provided in the following subsections, but those ideas only represent a few of the possible cross-curricular lessons. There are many online blogs and articles where teachers collaborate and share their own cross-curricular lesson ideas.

2.2.1 Integration with Math

Math/Science integration can be achieved by having students collect and record scientific data like temperatures, height of various plants, or the effects of gravity on different objects, then represent and interpret those data through the use of graphs [45]. Math/History-Social Science integration might require students to identify locations on a map and determine their latitudes and longitudes [44].

2.2.2 Integration with Language Arts

Language Arts/Science integration lessons may ask students to read a scientific paper or watch a scientific movie and then write journals, poems, or essays about what they have learned. Language Arts/History-Social Science integration can be accomplished by reading a book and relating it to the historical setting of the story. For example, students can read "Holes" by Louis Sachar [37] and relate the book to the historical context of racism.

2.2.3 Integration with Computer Science

Integration with Computer Science is no different than the integration with Math or Language Arts. Computer Science can be used to help with plotting graphs for Math, rendering different types of maps for History, and creating a blog web page with students' writing for Language Arts. The lessons developed for this thesis include these ideas, but there are many other possible lesson ideas to integrate Computer Science with other disciplines taught in school.

2.3 Platforms

Several web-based platforms are used for the interactive lab activities to demonstrate programming to students. The purpose of using different platforms is to introduce students to various tools that are available to them. Although the appearance of each platform looks slightly different, the underlying code that students get to modify is the same. This section discusses each of the platforms.

2.3.1 JSFiddle

JSFiddle is a web-based playground for HTML, CSS, and Javascript. The website allows users to customize their work environment, such as specifying the version of HTML and including frameworks to be used with Javascript. When the website is loaded, the user is presented with 4 panels. The top left panel is for HTML code, the top right panel is for CSS code, the bottom left panel is for JS (Javascript) code, and lastly the bottom right panel is for outputting the result. Figure 2.1 shows the user interface of JSFiddle. The toolbar at the top provides four functionalities, *Run*, *Save*, *Tidy*, and *Collaborate*. The important one for the students to note is *Run*. This button generates an output in the Result panel based on the code that is in the panels at the time the button is clicked.

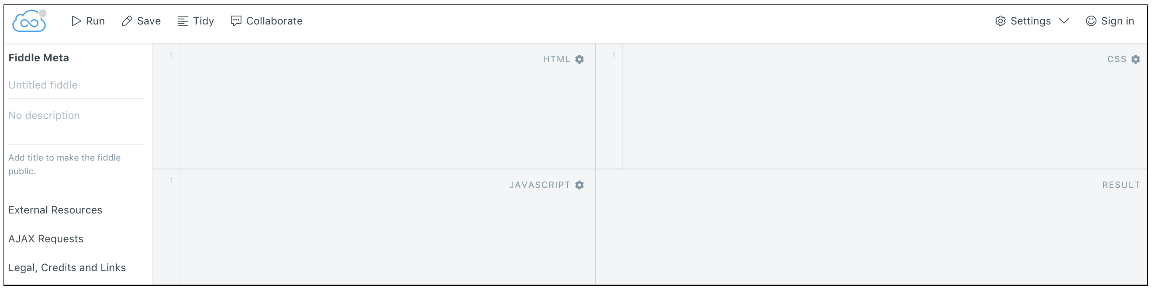


Figure 2.1: JSFiddle Interface

2.3.2 CodePen

CodePen is another web-based environment that allows users to design a website with HTML, CSS, and Javascript. What makes CodePen different from JSFiddle is that it also allows for testing, but it does not have as much Javascript library support. Unlike JSFiddle, CodePen updates the output automatically rather than having the user click a button to run the code. Besides those differences, the two editors are relatively similar, in that they both have the same 4 panels. CodePen serves as another platform for the students to work with in order to learn different tools. Figure 2.2 displays the user interface for CodePen.

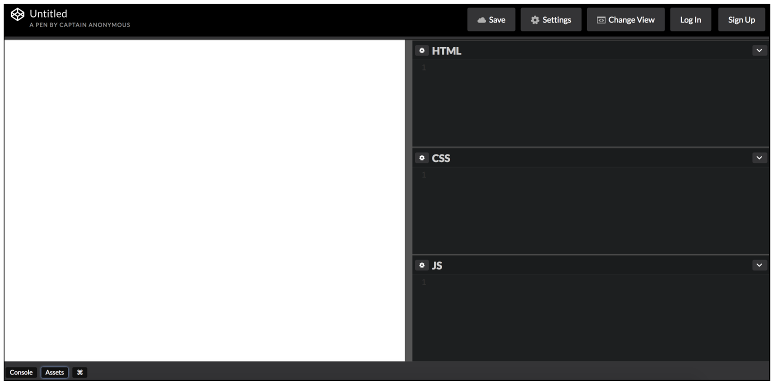
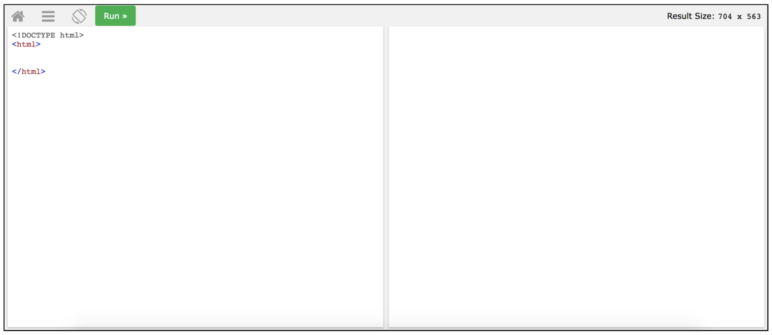


Figure 2.2: CodePen Interface

2.3.3 W3Schools

W3Schools is a website that provides tutorials on various web development technologies. Along with the tutorials, W3Schools provide a playground where users can edit and run their code to see the output. Its content includes SQL, PHP, Node.js, HTML, CSS, Javascript, and more. The user interface has two panels as displayed in Figure 2.3. The left panel is used for coding and the right panel is where the output is displayed once the *Run* button is clicked. W3Schools differs from JSFiddle and CodePen in that W3Schools does not have three different panels, but instead allows

HTML, CSS, and Javascript to be used in the same panel by using the <html>, <style>, and <script> tags respectively.



```
<!DOCTYPE html>
<html>


</html>
```

Figure 2.3: W3Schools Interface

Chapter 3

RELATED WORK

Others in the computer science and education communities have acknowledged the need to teach young children programming [1, 8]. Many tools are available today to teach computer science to children of all ages, such as board games [11, 12], card games [43], software programs [9, 24, 63], and hardware equipment [41]. Some of these tools are free while others cost money, upwards of $200. This section discusses some of the work that has been developed within the last decade.

3.1 Scratch

Scratch was publicly launched in 2013 as a free visual programming language to help young children learn coding by creating their "own interactive stories, games, and animations and share [their] creations with others in the online community" [59]. Scratch uses block programming, where every block represents a functionality such as Motion (moving), Data (variables), and Control (conditionals). Users can generate a program by putting together sequences of blocks. Scratch is used for K-12 schools but can also be utilized in some university classes.

Scratch provides various lessons to educators with a verified teacher account upon request. Subjects taught in school can be integrated with Scratch. For example, students can create a storyboard on a historical topic by typing up a short excerpt, and use sprites and animation to add movement in order to convey their story.

The primary difference between Scratch and this thesis is that Scratch uses a graphical programming approach to abstract the use of syntax while the lessons in this thesis expose students to text-based programming to introduce the more common

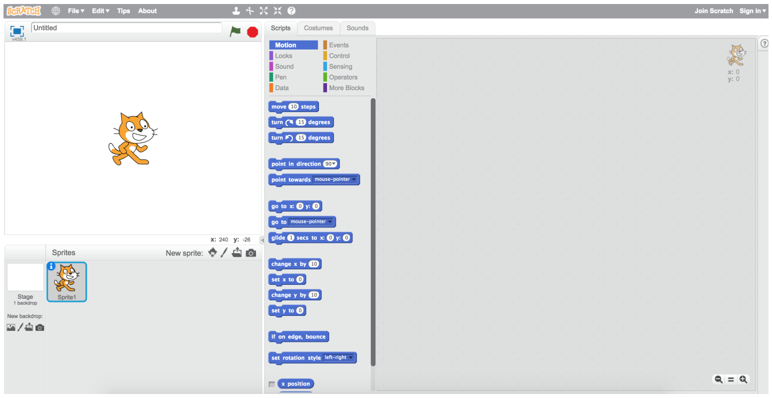


Figure 3.1: Scratch Interface

approach. Scratch aims to teach computer science concepts like loops, variables, and conditionals whereas this thesis work aims to teach the applications of computer science to spark interest in students. Another difference is that Scratch provides a blank canvas for students to work with while this thesis work provides code templates.

3.2 Code.org

Code.org is a non-profit that began in 2013 with a goal of "expanding access to computer science in schools and increasing participation by women and underrepresented minorities" [10]. Comparing to Scratch, Code.org also uses graphical programming and can also be used for K-12 classrooms as well as university classrooms. In contrast, Code.org activities present a puzzle and provide code blocks that are relevant to accomplish that task, whereas Scratch provides all available blocks and allows students to create and design.

Code.org provides Computer Science course catalogs for K-12 educators and also aims to teach concepts similar to Scratch. It also has "Hour of Code" activities that students can accomplish in an hour, such as making a Flappy Bird game or solving individual puzzles in settings that capture the students' interest (e.g. Minecraft and Star Wars). Code.org primarily teaches students computer science concepts with no

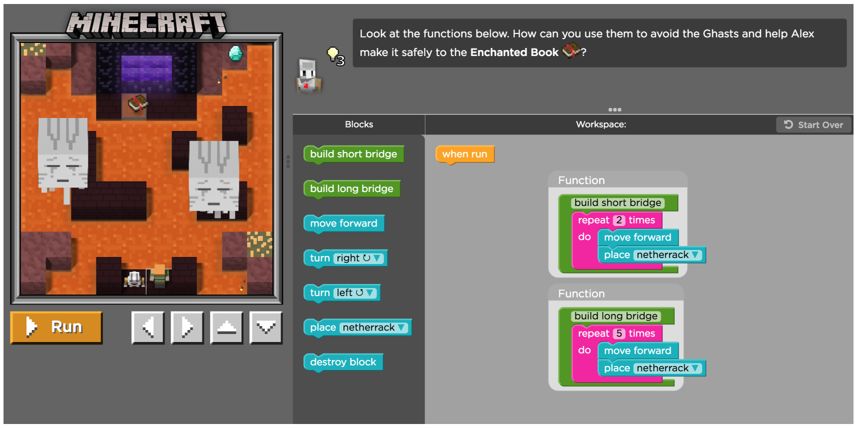

Look at the functions below. How can you use them to avoid the Ghasts and help Alex make it safely to the **Enchanted Book**?

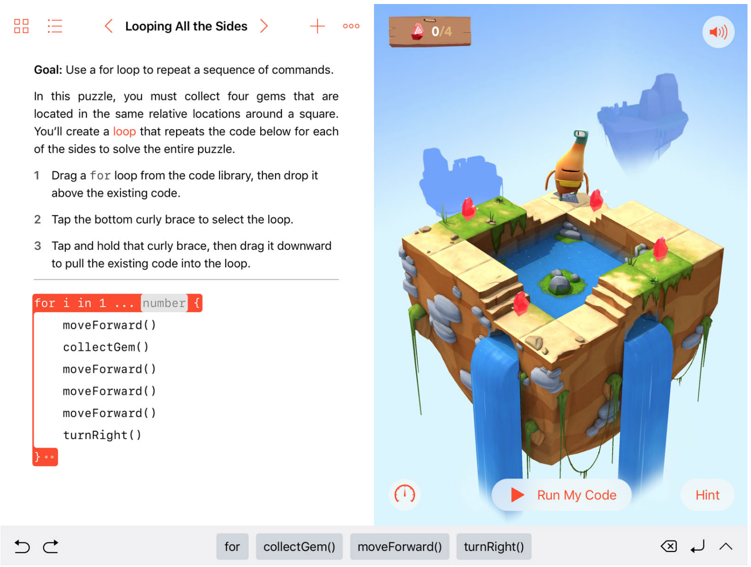
Figure 3.2: Code.org Interface

integration with other subjects.

3.3 Swift Playgrounds

Swift Playgrounds is an iPad app that was released in 2016 to target teaching young kids basic coding in Swift [60]. Swift Playgrounds is a puzzle-solving program with different difficulty levels that teaches computer science concepts like Code.org, but it is much more animated. Unlike Scratch and Code.org, Swift Playgrounds teaches a programming language, Swift, and is not based on block programming. Swift Playgrounds can be used in classrooms since many lessons are designed to be 45-60 minutes long, but the lessons are not cross-curricular. The app includes a special keyboard that provides shortcuts to make it easier to enter code. Swift Playgrounds can also be used to pair up and control compatible robots and drones, such as Lego Mindstorms, Spheros, and Parrot Drones, to name a few.

Although Swift Playgrounds is a free app that can teach anyone how to code in Swift through solving puzzles, the more interactive robots and drones cost money. There are many kits that are on the affordable side ranging between $50 to $200, but there are also extremely expensive ones that sell for thousands of dollars [61].



Goal: Use a for loop to repeat a sequence of commands.

In this puzzle, you must collect four gems that are located in the same relative locations around a square. You'll create a loop that repeats the code below for each of the sides to solve the entire puzzle.

1 Drag a `for` loop from the code library, then drop it above the existing code.

2 Tap the bottom curly brace to select the loop.

3 Tap and hold that curly brace, then drag it downward to pull the existing code into the loop.

```
for i in 1 ... number {
    moveForward()
    collectGem()
    moveForward()
    moveForward()
    moveForward()
    turnRight()
}
```

Figure 3.3: Swift Playgrounds Interface

To support classroom instruction at a minimum, there would need to be a sufficient number of iPads to accommodate all students.

3.4 Dash and Dot

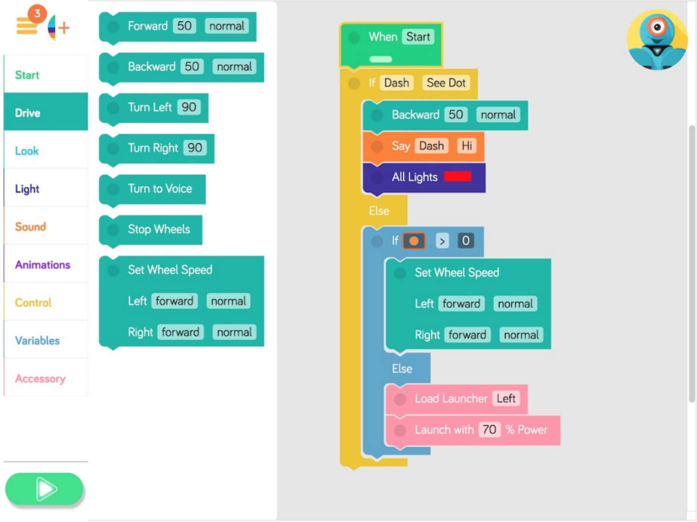
Dash and Dot are robots that were released in 2014 by Wonder Workshop. Dash, displayed in Figure 3.4, is a robot that can be programmed to navigate, dance, sing, and respond to voices. Dot, shown in Figure 3.5, is an accessory that is meant to be Dash's companion with less functionality. There are many iOS and Android apps that can be used to program Dash, such as Go, Blockly, and even Swift Playgrounds [21]. Figure 3.6 shows the interface of Blockly, one of the apps that is compatible with controlling Dash.

Educators can sign up for an account with Wonder Workshop to obtain cross-curricular lesson plans for subject areas like Math, Science, Social-Science, Art, and Language Arts [66]. Although these robots are incredibly interactive and may generate a lot of interest from students, they are expensive. Dot costs $49.99 and Dash

Figure 3.4: Dash

Figure 3.5: Dot

costs $149.99. Not only that, a tablet is also required to run the programming apps. If a classroom were to invest in these lesson plans, they would have to get enough robots and tablets to be efficiently shared among all of the students.



```
When Start
If Dash See Dot
    Backward 50 normal
    Say Dash Hi
    All Lights
Else
    If ● > 0
        Set Wheel Speed
        Left forward normal
        Right forward normal
    Else
        Load Launcher Left
        Launch with 70 % Power
```

Figure 3.6: Blockly Interface

Chapter 4

CONSIDERATIONS

4.1 Overview

Computer science may appear daunting to those with little knowledge about it, which can dissuade someone from pursuing the field. To combat this representation of computer science, exposing students to this academic area at an early age may result in a stronger desire to investigate the field further. However, requests that teachers include computer science in their classroom could face some push back if they are not comfortable using the necessary technology. The questions to ask here are: what can we do to help teachers feel more comfortable incorporating something they may not have confidence teaching and what should we highlight in the lessons to convey to students that computer science is applicable and accessible?

Planning out the scope of this task requires careful consideration to determine the process to move forward. This includes aspects such as identifying student age range, determining appropriate classroom materials, and surveying teachers' and students' needs. Furthermore, classroom resources must also be considered, such as time, available technology, and teacher technical skills. Once those factors are accounted for, development begins.

This section discusses some of the important considerations when aiming to create interactive computer science lessons for the classroom and a useful tool to aid teachers with integrating said lessons.

4.2 Student Target

The primary target for the computer science unit is elementary school students. In 2014, standardized testing converted from paper-based to online-based Common Core testing; this testing begins in third grade. Although that implies that third grade students are computer literate, based on my previous volunteering experience it seems like the students begin to become efficient at using a computer and more comfortable with typing starting in the fourth grade. Since the lessons developed in this thesis work require navigating between websites on a computer and some typing for programming, the minimum grade level is determined to be fourth grade. Many elementary schools could go up to the sixth grade, which is the case for the participating elementary schools in this study. Thus the target group is decided to be the upper grades of elementary school, ranging from fourth to sixth grade.

4.3 Teacher Input

Given that one of the main goals is to have teachers adopt the computer science unit developed in this thesis work, getting input from teachers is necessary to understand what will most likely encourage them to integrate the lessons into their classroom.

Based on the different feedback received from several teachers, the general concern for them is that they do not have enough time in the school year to stray away from the curriculum. Many of the teachers claim that they would be more inclined to adopt the computer science unit if the activities were aligned with the standards. They also indicate that they would be much more comfortable teaching those lessons if there were a tool to help them.

4.4 Time Constraints

According to input from different teachers, it is typical for a lesson to be around an hour long. The first half of instruction is spent on introducing new material to the students and the remaining time is allotted for the students to practice what they have learned. If the lessons in the computer science unit are aimed to be 50-minutes long, they could potentially be adopted by middle school teachers since each period is approximately 50 minutes.

The lessons are structured based on the Five-Step Lesson approach [42]. The first half of the period is spent on introductions and presentation of the new materials to convey the applications of computer science. The second half is allocated for the students to apply what they have learned by altering code and making observations about the result. Since the students would not be able to type out a complete meaningful program from scratch in such a short amount of time, it makes sense to provide most of the source code so that they would only have to fill in the missing pieces.

4.5 Classroom Resources

There are many existing interactive activities that can be used in the classroom, such as Dash and Dots, Spheros, and Lego Mindstorms, but the disadvantage of these tools is their cost (as discussed in Chapter 2). In order for the computer science unit to be adopted into the classroom, affordability and accessibility are important. The only equipment needed for the lessons presented in this thesis are computers for the students to program on and a projector for the teacher to demonstrate the lesson. The requirement for computers should not be a problem since a large percentage of public schools now have computers with internet access for students to use. According to the National Center for Education Statistics (NCES) in 2008, 100% of public elementary schools have computers with internet access with a ratio of students to computers of 3.2 and 55% of public elementary schools have laptop computers on carts in the classroom [50].

4.6 Web Development

There are many online code editors for web development that are convenient to use and do not require any installations. The only requirement necessary to access the code editors is a web browser. Web development activities make demonstrating computer science simple due to its visual components. Allowing students to create a product that they like and to be able to see the results immediately could help captivate their interest in computer science. Source codes provided in the computer science unit are written in Javascript, HTML, and CSS, and students are given a chance to work with each of the three languages.

4.7 Technical Skills

Many teachers do not have extensive programming knowledge, thus a tool created for them must be simple to use with detailed instructions. Directions should be broken down into smaller steps in order to be explicit of what is expected. Images should be included wherever appropriate to ensure that teachers are confident that they are following the instructions correctly.

Chapter 5

INTEGRATED COMPUTER SCIENCE UNIT

The lessons in the computer science unit integrate Computer Science with other disciplines, such as Language Arts, Mathematics, and History-Social Science. The Language Arts and Mathematics standards that were used in this thesis work follow the Common Core State Standards (CCSS), the Science standards follow California's Next Generation Science Standards (NGSS), and the History-Social Science standards follow California's History-Social Science Standards (HSS).

5.1 Participants

The participants for this study are 4th grade students from Grover Heights Elementary School, 5th grade students from Lange Elementary School, and a mixture of 4th and 5th grade students from Pacheco Elementary School.

5.1.1 Grover Heights Elementary School

Grover Heights Elementary School is a public school located in Grover Beach in San Luis Obispo County with a student-to-teacher ratio of 22:1. Out of the 445 students enrolled in the school, 63 students are in 4th grade, 240 students identify themselves as White, and 161 students consider themselves as Hispanic [33]. 9.4% of the students are English learners and 49.4% of the students are on free or reduced meals [34].

The California Department of Education collects data and reports on each school's performance on standardized tests, called *California Assessment of Student Performance and Progress (CAASPP)*. At an elementary school, the test is given to 3rd through 6th grade students. For the report based on the 2015-2016 school year at

Grover Heights Elementary School, 54% of the students exceeded or met the English standard and 40% of the students exceeded or met the Mathematics standard [57].

There are 26 students in the particular classroom in which the lessons were presented. Each session takes place in the school's computer lab. The sessions are 50 minutes long and occur once a week for four weeks.

5.1.2 Lange Elementary School

Lange Elementary School is a public school located in Nipomo in San Luis Obispo County with a student-to-teacher ratio of 26:1. The school consists of 564 students, where 223 identify themselves as White and 307 identify themselves as Hispanic. The 5th grade class is made up of 93 students [39]. 20.1% of the students are English learners and 54.7% of the students are on free or reduced meals [40].

The results of the *CAASPP* for the 2015-2016 school year at Lange Elementary School indicated that 48% of the students met or exceeded the English standard and 38% of the students met or exceeded the Mathematics standard [57].

Each session is 50 minutes long and takes place in the provided classroom with 32 students, where the students have individual Chromebooks to work on.

5.1.3 Pacheco Elementary School

Pacheco Elementary School is a bilingual public school located in San Luis Obispo with a student-to-teacher ratio of 24:1. The total number of students at the school is 541, with 195 identifying themselves as White and 309 identifying themselves as Hispanic. 81 of the students are in 4th grade, 66 of the students are in 5th grade, and 59 of the students are in 6th grade [51]. 44.1% are English learners and 44.1% are on free or reduced meals [52].

For the *CAASPP* for the 2015-2016 school year at Pacheco Elementary School, 60% of the students met or exceeded the English standard and 56% of the students met or exceeded the Mathematics standard [57].

The sessions are held as an after school program called Pacheco Code Club. The Code Club takes place for 4 weeks every academic quarter except Summer. The program is led by a parent, who is also a lecturer at Cal Poly, and usually follows the Code.org curriculum but was replaced with this thesis work for one quarter. Each session is 50 minutes long and takes place in the computer lab with 15 students.

5.2 Short Lectures

Each session starts with a discussion about basic computer science vocabulary and a short lecture (roughly 20 minutes) to prepare the students for the lab portion. The goals of the lectures are to introduce students to the ways programming can be tied to the topic of the day and to give them enough knowledge so that they can work on the lab portion. The lessons are a mixture of instructional time, demonstrations, and interactive opportunities. All programming demonstrations are projected onto a screen displayed in front of the class. The topics included in this thesis work are Geography, Writing, Graphs, and California Missions. Time is closely managed during the lecture in order to ensure that enough time is left over for the lab activity.

5.2.1 Lecture: Geography

This lesson demonstrates a walk-through of how computer science can generate helpful and meaningful maps. The purpose is to reveal to students that coding can help with visualizing and representing information, personalizing data, and altering the source code will change the content and style of the information displayed. Table 5.1 outlines the agenda for this lesson.

Since this is the first lesson, students are asked to take the pre-survey with questions regarding their knowledge and interest in computer science. The pre-survey takes place in the very beginning of the session, before any introductions of the lesson material. Further details about the pre-survey are discussed in Chapter 7.

Table 5.1: Agenda for Geography Lesson

Minutes	**Geography Lecture**
10	Pre-survey
5	Vocabulary discussion
2	Overview of lesson and purpose of maps
3	Introduction to Google Spreadsheet and how to enter data
5	How to change colors of the map
	Geography Lab
10	Input population data into Google spreadsheet
10	Change map colors, observe dark and light colored states
	Geography Discussion
5	Wrap up questions and discussion

5.2.2 Lecture: Writing

This lesson shows how programming is the basis for creating web pages. The lesson shows students how to do research on the internet in order to obtain information and images that they want for their website. The lecture also demonstrates how a web page can be customized by changing the content and style. The goal is to convey to the students that web development is broadly used and computer science is a huge part in making it possible. The agenda for this lesson is outlined in Table 5.2.

Table 5.2: Agenda for Writing Lesson

Minutes	Writing Lecture
5	Vocabulary discussion
5	Overview of web development and applications: blogs & websites
3	How to research for information and images online
5	How to add images, change texts, colors, and fonts for a website
	Writing Lab
5	Find image and get its address
15	Research about topic and type a paragraph
7	Change colors and fonts of website
	Writing Discussion
5	Wrap up questions and discussion

5.2.3 Lecture: Graphs

This lesson teaches students about different types of graphs, such as line graphs and pie graphs. The goals are to show students how to graph their own data using programming and how to interpret them in order to obtain meaningful information. The lesson briefly explains what functions are in programming and how to call them. Table 5.3 displays the agenda for this lesson.

5.2.4 Lecture: California Missions

This lesson explains what an application programming interface (API) is and how APIs are relevant in computer science. The example API used in this lesson is the Google Maps API. The demonstration will show students how they can take advantage of the provided functionality and use it for different purposes than just navigation.

Table 5.3: Agenda for Graphs Lesson

Minutes	Graphs Lecture
5	Vocabulary discussion
3	Overview of lesson and different types of graphs
3	Introduction to functions in programming & how to call functions
5	How to add data to graphs, customize colors and texts of graphs
	Graphs Lecture
5	Input data into Google spreadsheet
5	Call function to plot line graph and make observations
3	Call function to plot pie graph
10	Change categories of slices and respective numbers
6	Change colors of slices
	Graphs Discussion
5	Wrap up questions and discussion

Table 5.4 shows the agenda for this lesson.

Since this is the last lesson in the unit, students are asked to take a post-survey in order to obtain data to compare to the pre-survey. The post-survey is given out to students at the end of the lab time. Further details about the surveys are discussed in Chapter 7.

5.3 Lab Details

The lab time (approximately 20-30 minutes) is reserved for students to begin their coding and apply what they have learned from the lecture (the allotted lab time seems to be sufficient for students to finish the lab activity, although many students generally needed the whole time). There are optional tasks beyond the lab guide for

Table 5.4: Agenda for California Missions Lesson

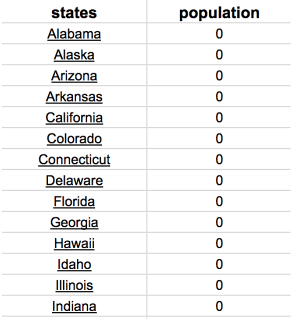
Minutes	California Missions Lecture
5	Vocabulary discussion
3	Overview of California missions
2	Introduction to Google Maps API
5	How to add locations onto map using Google Maps API
2	How to navigate and obtain directions between two locations
	California Missions Lab
5	Adjust initial longitude and latitude so that map centers in on San Luis Obispo, California
3	Copy and paste missions for dropdown mission list
4	Explore getting directions between different missions and different transportation methods
6	Customize map by changing colors of mission markers
	California Missions Discussion
5	Wrap up questions and discussion
10	Post-survey

students that do finish early.

All necessary materials and links are posted on the teacher's Google Classroom [30] account to which the students have access. As the students are working on the activity during lab time, instructions from the lecture are broken down again step by step. Students usually complete tasks and move along the lab as a class but occasionally a few students can fall behind the expected schedule. If that is the case, they are prioritized to receive help, either from their neighbor, teacher, or researcher. The lab wraps up with a few questions regarding their understanding depending on the standards for their grade level.

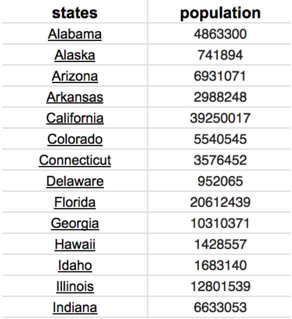
5.3.1 Lab: Geography

For this lab, 50 wooden craft sticks are prepared with the names of the 50 states of the United States on them. Students will receive either 1 or 2 states depending on the number of students in the classroom, and their first task is to research the population for the states they received. Once everyone has found the population for their state(s), the next step is for the students to enter their population into a shared Google spreadsheet. Figure 5.1 shows the original spreadsheet that is given to students and Figure 5.2 shows how the spreadsheet might look once students have filled in their population.



states	population
Alabama	0
Alaska	0
Arizona	0
Arkansas	0
California	0
Colorado	0
Connecticut	0
Delaware	0
Florida	0
Georgia	0
Hawaii	0
Idaho	0
Illinois	0
Indiana	0

Figure 5.1: Original spreadsheet



states	population
Alabama	4863300
Alaska	741894
Arizona	6931071
Arkansas	2988248
California	39250017
Colorado	5540545
Connecticut	3576452
Delaware	952065
Florida	20612439
Georgia	10310371
Hawaii	1428557
Idaho	1683140
Illinois	12801539
Indiana	6633053

Figure 5.2: Finished spreadsheet

The map starts out with a single color as displayed in Figure 5.3, but once the Google spreadsheet has all the data, students can click *Run* in JSFiddle (discussed in Section 2.3.1) to observe that their map is now color coded as in Figure 5.4. Finally, the students get to customize their map by adding a title and subtitle to their map and changing the colors using HTML and hex color codes.

In order to change the title and subtitle, students are instructed to go to the appropriate line numbers and insert the text within the single quotes. As shown in Listing 5.1, students have to navigate to line 27 to fill in the text for the map title

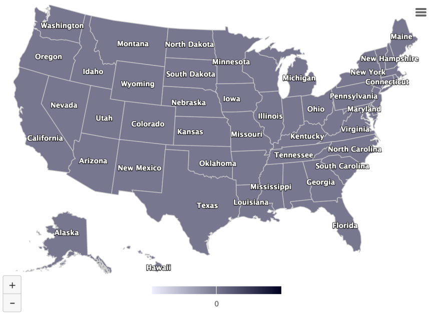


Figure 5.3: Map without population

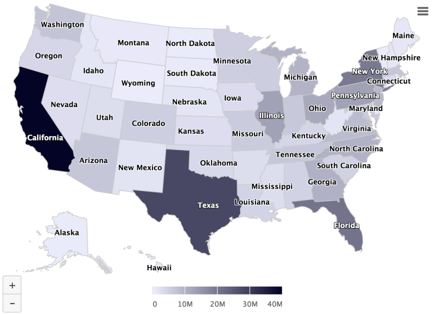


Figure 5.4: Map with population

and line 31 to fill in the text for the map subtitle. Students are told to use "United States of America" as the map title and an informative description for the subtitle.

Listing 5.1: Original Javascript Code Snippet for Geography Lab

```
25 title: {
26     // Insert text for title between single quotes
27     text: ''
28 },
29 subtitle: {
30     // Insert text for subtitle between single quotes
31     text: ''
32 },
33 colorAxis: {
34     // Replace color code between single quotes below to change
           light color
35     minColor: '#EEEEFF',
36     // Replace color code between single quotes below to change dark
            color
37     maxColor: '#000022'
38 },
```

In order to change the color spectrum for the states, students are instructed to find two colors they want from the W3Schools online color picker [15], one for the light end of the spectrum and one for the dark end of the spectrum. Next, the students have to navigate to line 35 to replace the hex code with the code for the light color that they had chosen and repeat on line 37 with the code for the dark color.

At the end of lab, all students should have their own unique customized map with the colors of their choosing. Figure 5.5 shows an example of a finished map generated by the altered code displayed in Listing 5.2.

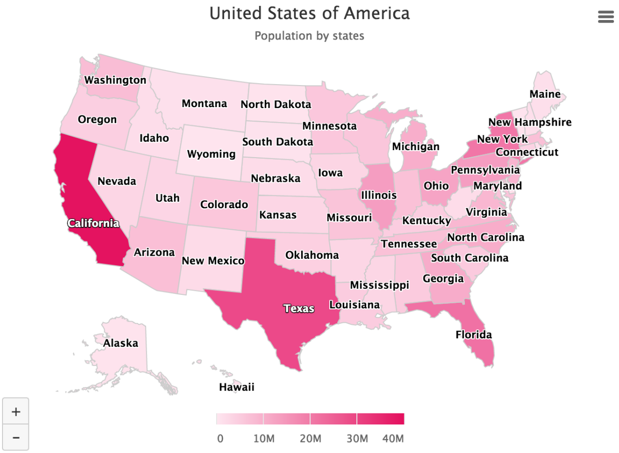


Figure 5.5: Sample of Geography Lab

Listing 5.2: Edited Javascript Code Snippet for Geography Lab

```
25 title: {
26     // Insert text for title between single quotes
27     text: 'United States of America'
28 },
29 subtitle: {
30     // Insert text for subtitle between single quotes
31     text: 'Population by states'
32 },
33 colorAxis: {
34     // Replace color code between single quotes below to change
          light color
35     minColor: '#ffe6f0',
36     // Replace color code between single quotes below to change dark
           color
```

```
    maxColor: '#e6005c'
},
```

5.3.2 Lab: Writing

This lab uses CodePen (discussed in Section 2.3.2) as the editor and the provided source code initially produces Figure 5.6. The tasks include gathering an image, researching a topic to write a paragraph about said topic, and customizing a web page. The topics are to be current topics the students are learning in class, but the specific topics are up to them to choose.

Listing 5.3: Original HTML Code Snippet for Writing Lab

```
12 <header>
13     <!-- Replace ‘My Topic’ with your title -->
14     <div class="title">My Topic</div>
15     <!-- Replace ‘Created by’ with your name -->
16     <h2>Created by</h2>
17 </header>
18 <!-- Replace the text between the double quotes below with your image URL -->
19 <img src="https://upload.wikimedia.org/wikipedia/commons/thumb/c/c9/
       -Insert_image_here-.svg/640px-Insert_image_here-.svg.png">
20 <div class="topicDetails">
21     <!-- Replace ‘Paragraph here’ with your paragraph -->
22     <p>Paragraph here about your topic here.</p>
23 </div>
```

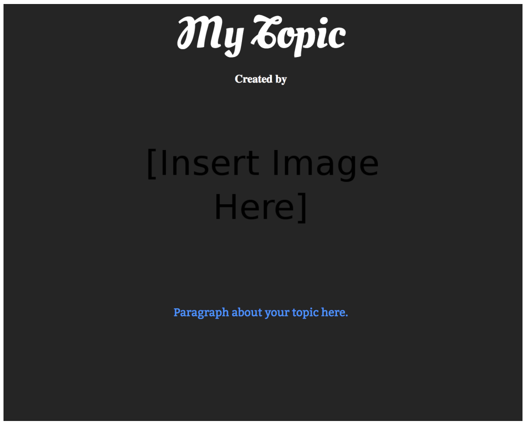
To start the lab, students are instructed to replace the default text of *My Topic* with the name of their topic and replace the default text of *Created by* with their own name. The changes can be made on lines 14 and 16 respectively shown in the HTML code displayed in Listing 5.3. Students are then taught to find an image on Google, copy its image address, and paste their image address to replace the default

URL between the double quotation marks on line 19 in the HTML code. After they have successfully displayed a picture, their next step is to find as much information about their topic and type up an informative paragraph on line 22 of the HTML code. Once the students are done with these steps, the format of their output should look similar to Figure 5.7, which is generated by the code in Listing 5.4.

Listing 5.4: Edited HTML Code Snippet for Writing Lab

```
12 <header>
13     <!-- Replace ‘Blue Whales’ with your title -->
14     <div class="title">Blue Whales</div>
15     <!-- Replace ‘Created by’ with your name -->
16     <h2>Di Hoang</h2>
17 </header>
18 <!-- Replace the text between the double quotes below with your image URL -->
19 <img src="https://www.techworm.net/wp-content/uploads/2017/08/
      blue-whale-wallpaper-27-1024x539.png">
20 <div class="topicDetails">
21     <!-- Replace ‘Paragraph here’ with your paragraph -->
22     <p>The blue whale is a marine mammal that is known to be the
          largest animal to have ever existed. They can reach up to 98
          feet in length and weigh up to 400,000 pounds. A blue whale
          can eat up to 400 million krills a day. The longest living
          blue whale was recorded to be 34 years old. If you want to go
           whale watching and spot a blue whale, they can be seen off
          the Southern California coast between April and August.</p>
23 </div>
```

After making changes to the content using HTML, students move on to change the style of their fonts and the color of their texts and background using CSS. Changing the colors is similar to the task in the Geography lab. Students can navigate to the color picker and replace hex color codes on lines 14, 16, 22, and 30 displayed in



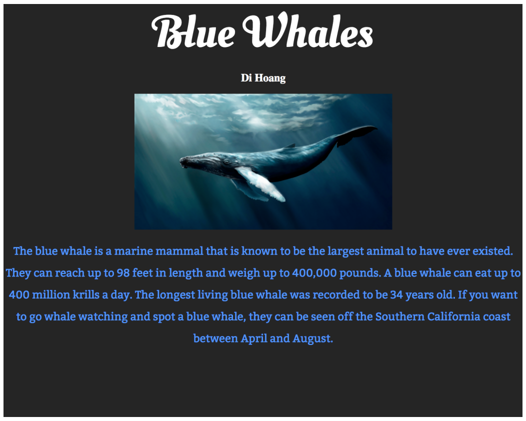
My Topic

Created by

[Insert Image Here]

Paragraph about your topic here.

Figure 5.6: Original website



Blue Whales

Di Hoang

The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day. The longest living blue whale was recorded to be 34 years old. If you want to go whale watching and spot a blue whale, they can be seen off the Southern California coast between April and August.

Figure 5.7: Website with content

Listing 5.5. Students are instructed to determine what each line changes by reading the comment directly above it.

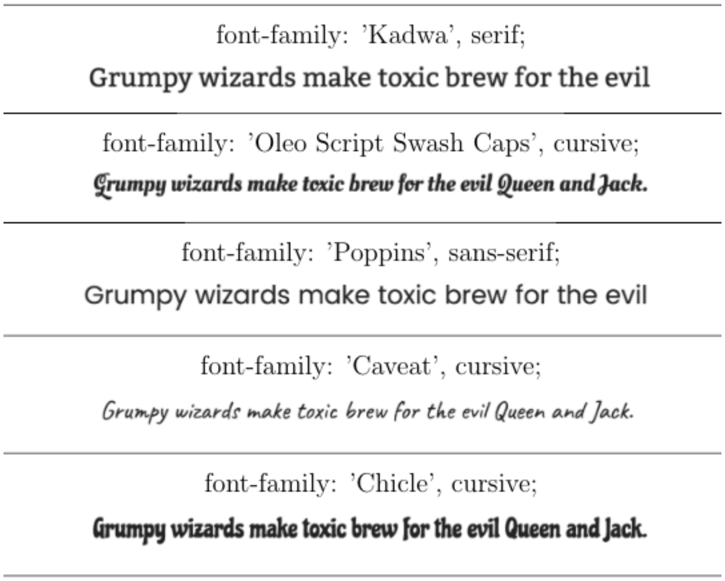
Listing 5.5: Original CSS Code Snippet for Writing Lab

```
body{
    /* change the color of your background below */
    background: #252525;
    /* change the color of your title below */
    color: #ffffff;
}
h2{
    text-align: center;
    font-size: 100%;
    /* change the color of your name below */
    color: #ffffff;
}
p{
    text-align: center;
    font-size: 100%;
    /* change the font of your paragraph below */
    font-family: 'Kadwa', serif;
    /* change the color of your paragraph below */
30     color: #4d94ff;
31 }
```

To change the fonts, students have to choose a font that they want from a list of font styles displayed in Figure 5.8, copy the code of their desired font, and replace the entire line of CSS code on line 28 with the new font they copied. Figure 5.9 displays the final product after the paragraph font, paragraph color, title color, and author name color have been changed.

Listing 5.6: Edited CSS Code Snippet for Writing Lab

```
12 body {
13     /* change the color of your background below */
14     background: #ffffff;
15     /* change the color of your title below */
16     color: #66ff99;
17 }
18 h2 {
19     text-align: center;
20     font-size: 100%;
21     /* change the color of your name below */
22     color: #3366ff;
23 }
24 p {
25     text-align: center;
26     font-size: 100%;
27     /* change the font of your paragraph below */
28     font-family: 'Caveat', cursive;
29     /* change the color of your paragraph below */
30     color: #ff6699;
31 }
```



font-family: ’Kadwa’, serif;

Grumpy wizards make toxic brew for the evil

font-family: ’Oleo Script Swash Caps’, cursive;

Grumpy wizards make toxic brew for the evil Queen and Jack.

font-family: ’Poppins’, sans-serif;

Grumpy wizards make toxic brew for the evil

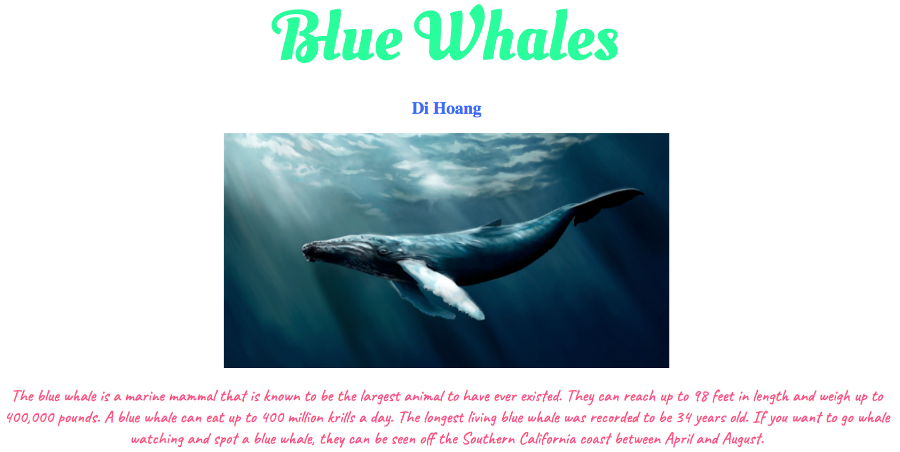
font-family: ’Caveat’, cursive;

Grumpy wizards make toxic brew for the evil Queen and Jack.

font-family: ’Chicle’, cursive;

Grumpy wizards make toxic brew for the evil Queen and Jack.

Figure 5.8: List of Fonts



Blue Whales

Di Hoang

The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day. The longest living blue whale was recorded to be 34 years old. If you want to go whale watching and spot a blue whale, they can be seen off the Southern California coast between April and August.

Figure 5.9: Sample of Writing Lab

5.3.3 Lab: Graphs

This lab teaches students about line graphs and pie graphs, allowing them to add data to the graphs, and to customize the graphs. When students first navigate to JSFiddle (discussed in Section 2.3.1), the result panel will be blank. The line graph will not show until students provide the URL of the shared spreadsheet that will be used for entering data. The pie graph will not be displayed until students call the function to plot the pie graph.

To display the coordinate system for the line graph, students have to copy the URL of the provided spreadsheet and paste it into the Javascript code between the single quotation marks on line 14 in Listing 5.7 to obtain the code in Listing 5.8. Once students click *Run*, a coordinate system with no data points will appear as shown in Figure 5.10. To obtain data points for the line graph, students have to work together. Students are assigned a day number, starting at 1. They have to find the temperature of their day in January 2018 from Weather Underground and input the number into a spreadsheet. The process of inputting data into a spreadsheet is the same as the Geography lesson. If there are fewer students than the number of days in the given month, the researcher enters in the data for the remaining days. Once the spreadsheet is completed, students can click the *Run* button in JSFiddle and see that their line graph now has data points as shown in Figure 5.11.

Listing 5.7: Original Javascript Code Snippet for Graphs Lab

```
 4  function drawCharts() {
 5    /* insert Google spreadsheet between single quotes */
 6      var query1 = new google.visualization.Query('');
 7      query1.send(lineGraph);
 8      /* call pieGraph(); function below */
 9
10  }
```

Listing 5.8: Edited Javascript Code Snippet for Line Graph

```
 4  function drawCharts() {
 5    /* insert Google spreadsheet between single quotes */
 6      var query1 = new google.visualization.Query('https://
          docs.google.com/spreadsheets/d/19
          NvbEFthSPRG6WFdt0OPBgfICuS8SWfBmq9Mfe5BWZ8/edit#gid=0');
```

```
7       query1.send(lineGraph);
8       /* call pieGraph(); function below */
9
10  }
```

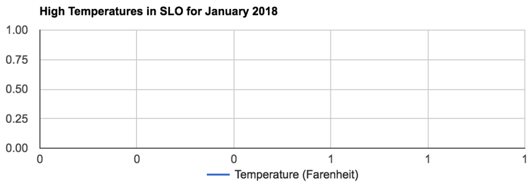


Figure 5.10: Graph without data

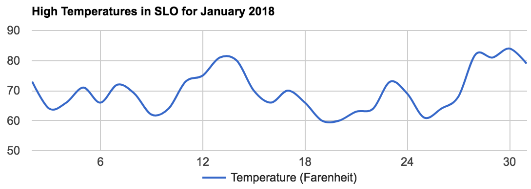


Figure 5.11: Graph with data

Moving onto the pie graph, students will have to call the function to plot the pie graph on line 9 of the Javascript code as shown in Listing 5.9. Students will get to customize the activities to reflect their own daily routine, the number of hours they spend each day on the activities, and the colors of each slice in the pie graph.

Listing 5.9: Edited Javascript Code Snippet to Call Pie Graph

```
4   function drawCharts() {
5     /* insert Google spreadsheet between single quotes */
6       var query1 = new google.visualization.Query('https://
           docs.google.com/spreadsheets/d/19
           NvbEFthSPRG6WFdt0OPBgfICuS8SWfBmq9Mfe5BWZ8/edit#gid=0');
7       query1.send(lineGraph);
8       /* call pieGraph(); function below */
9       pieGraph();
10  }
```

After students are able to render the pie graph, their next task is to change the daily activities and the number of hours they spend on each activity. Listing 5.10 shows the Javascript code that determines how the pie graph in Figure 5.12 is distributed and the color of each slice.

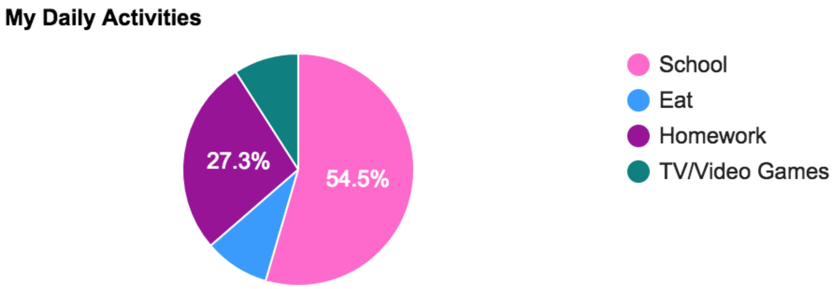


Figure 5.12: Original Pie Graph

Listing 5.10: Original Javascript Code for Pie Graph Activities & Hours

```
38 function pieGraph() {
39   var data = google.visualization.arrayToDataTable([
40     ['Task', 'Hours per Day'],
41     ['School', 6],
42     ['Eat', 1],
43     ['Homework', 3],
44     ['TV/Video Games', 1]
45   ]);
46   var options = {
47     title: 'My Daily Activities',
48     slices: {
49       /* Fill in color codes here */
50       0: { color: '#ff66cc' },
51       1: { color: '#3399ff' },
52       2: { color: '#990099' },
53       3: { color: '#008080' }
54     }
55   };
56 }
```

To change the daily tasks, students can replace the task names inside the single quotation marks on lines 41 through 44. Similarly, students can change the number of hours they spend on each task by altering the numbers on the same lines. The number of hours can be formatted as an integer or decimal. To change the colors of the slices, the task is similar to changing hex color codes from previous labs. Students can accomplish this by replacing the hex color codes inside the single quotation marks on lines 50 through 53. The hex color codes are ordered in the same manner as the tasks, which means the color code on line 50 corresponds to the task on line 41, the color code on line 51 corresponds to the task on line 42, so on so forth. After changes are made as shown in Listing 5.11, the resulting pie graph should look like Figure 5.13.

Listing 5.11: Edited Javascript Code for Pie Graph Activities & Hours

```
38 function pieGraph() {
39   var data = google.visualization.arrayToDataTable([
40     ['Task', 'Hours per Day'],
41     ['School', 7],
42     ['Eat', 1],
43     ['Sports', 1.5],
44     ['Reading', 3]
45   ]);
46   var options = {
47     title: 'My Daily Activities',
48     slices: {
49       /* Fill in color codes here */
50       0: { color: '#00ff99' },
51       1: { color: '#4040bf' },
52       2: { color: '#ffff99' },
53       3: { color: '#ffa64d' }
54     }
55   };
56 }
```

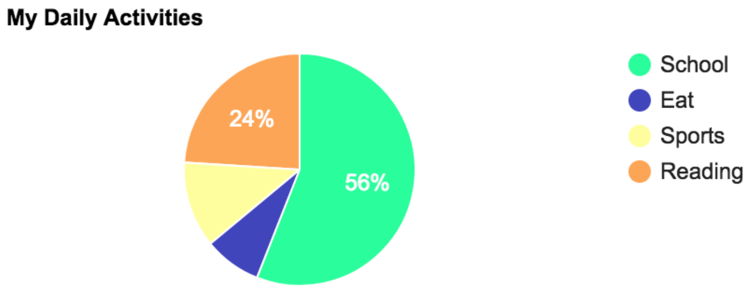


Figure 5.13: Sample of Graphs Lab

5.3.4 Lab: California Missions

This lab includes teaching students about the Google Maps API and using it to display some of the California missions using the W3Schools editor (discussed in Section 2.3.3). The purpose of the lab is to help students visualize the missions along the coastline and find out how long it would take to travel between the missions via various transportation methods.

Listing 5.12: Javascript Code for Original Coordinates

```
function initMap() {
    var directionsService = new google.maps.DirectionsService;
    var directionsDisplay = new google.maps.DirectionsRenderer;
    var map = new google.maps.Map(document.getElementById('map'), {
      zoom: 7,
      center: {lat: 50.5, lng: -100.5}
    });
    directionsDisplay.setMap(map);
}
```

The original code produces a map that zooms in on an arbitrary location as shown in Figure 5.14. The first task is for students to get the map to focus in on San Luis Obispo, California. In order to do so, students need to look up the latitude and longitude of the city and replace the original coordinates with the new coordinates on line 64 of the Javascript code shown in Listing 5.12. The latitude and longitude parameters take positive numbers for North and East, and negative numbers for South and West. Once students change their code to look like Listing 5.13 and click the *Run* button, they should be able to see San Luis Obispo and several plotted California missions as shown in Figure 5.15.

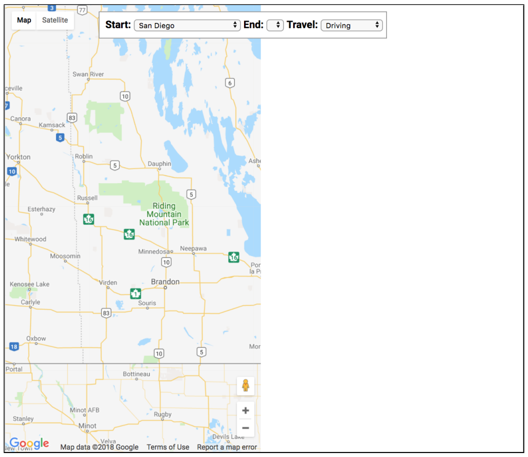


Figure 5.14: Original Coordinates

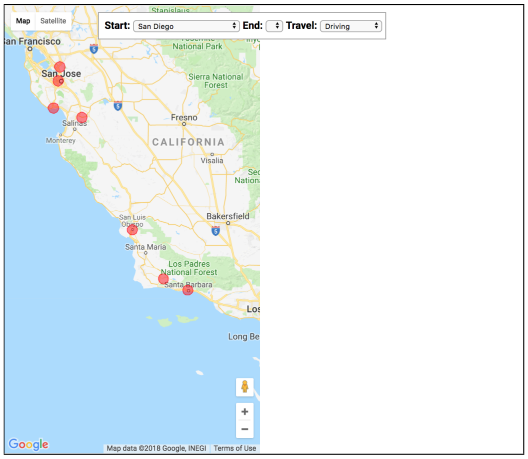


Figure 5.15: New Coordinates

Listing 5.13: Javascript Code for Coordinates of San Luis Obispo

```
59 function initMap() {
60     var directionsService = new google.maps.DirectionsService;
61     var directionsDisplay = new google.maps.DirectionsRenderer;
62     var map = new google.maps.Map(document.getElementById('map'), {
63       zoom: 7,
64       center: {lat: 35.28, lng: -120.66}
65     });
66     directionsDisplay.setMap(map);
67 }
```

Next, students are asked to observe that they cannot navigate for directions yet because the destinations are missing. The original HTML code for the *Start* and *End* dropdown lists is shown in Listing 5.14. Students are to copy the code for the *Start* dropdown list from lines 26 through 34 and paste it onto line 40 to create the *End* dropdown list.[1] Once students accomplish the task, their code should look like the code in Listing 5.15. Once the *Run* button is clicked, students will be able to select the *Start* and *End* locations to obtain directions. Figure 5.16 shows an example of the directions from Mission San Diego de Alcalá in San Diego to Mission San José in Fremont. Students are asked to use different methods of travel, compare the time it took people in the past to travel by walking versus how fast we can travel today with cars, and note the importance of technology.

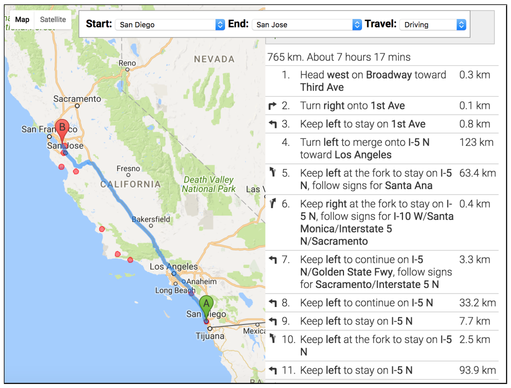
Listing 5.14: Original HTML Code for Dropdown Lists

```
23 <b>Start:</b>
24 <select id="start">
25   <!-- HIGHLIGHT STARTING FROM THE LINE BELOW -->
26   <option value="san diego, ca">San Diego</option>
27   <option value="san capistrano, ca">San Juan Capistrano</option>
28   <option value="santa barbara, ca">Santa Barbara</option>
29   <option value="solvang, ca">Santa Ines</option>
30   <option value="san luis obispo, ca">San Luis Obispo de Tolosa</option>
31   <option value="san bautista, ca">San Juan Bautista</option>
32   <option value="santa cruz, ca">Santa Cruz</option>
33   <option value="santa clara, ca">Santa Clara de Asis</option>
34   <option value="fremont, ca">San Jose</option>
35   <!-- UP TO THE LINE ABOVE, THEN COPY -->
36 </select>
37 <b>End:</b>
38 <select id="end">
39   <!-- PASTE THE CODE BELOW -->
40
41 </select>
```

[1] Initially, students were to research the missions and add the locations to the dropdown list accordingly. But in the interest of time, students were asked to copy and paste instead.

Listing 5.15: Edited HTML Code for Dropdown Lists

```html
<b>Start:</b>
<select id="start">
    <!-- HIGHLIGHT STARTING FROM THE LINE BELOW -->
    <option value="san diego , ca">San Diego</option>
    <option value="san capistrano , ca">San Juan Capistrano</option>
    <option value="santa barbara , ca">Santa Barbara</option>
    <option value="solvang , ca">Santa Ines</option>
    <option value="san luis obispo , ca">San Luis Obispo de Tolosa</option>
    <option value="san bautista , ca">San Juan Bautista</option>
    <option value="santa cruz , ca">Santa Cruz</option>
    <option value="santa clara , ca">Santa Clara de Asis</option>
    <option value="fremont , ca">San Jose</option>
    <!-- UP TO THE LINE ABOVE, THEN COPY -->
</select>
<b>End:</b>
<select id="end">
    <!-- PASTE THE CODE BELOW -->
    <option value="san diego , ca">San Diego</option>
    <option value="san capistrano , ca">San Juan Capistrano</option>
    <option value="santa barbara , ca">Santa Barbara</option>
    <option value="solvang , ca">Santa Ines</option>
    <option value="san luis obispo , ca">San Luis Obispo de Tolosa</option>
    <option value="san bautista , ca">San Juan Bautista</option>
    <option value="santa cruz , ca">Santa Cruz</option>
    <option value="santa clara , ca">Santa Clara de Asis</option>
    <option value="fremont , ca">San Jose</option>
</select>
```

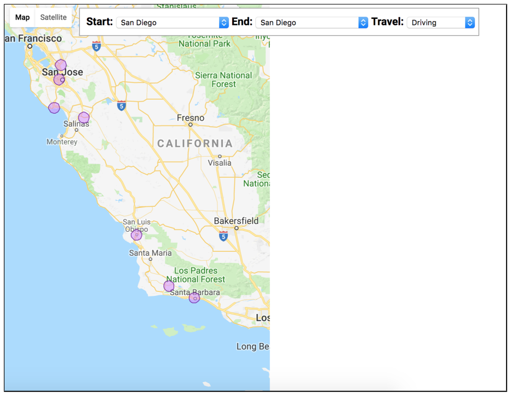


Start: San Diego **End:** San Jose **Travel:** Driving

765 km. About 7 hours 17 mins

1. Head **west** on **Broadway** toward **Third Ave** 0.3 km
2. Turn **right** onto **1st Ave** 0.1 km
3. Keep **left** to stay on **1st Ave** 0.8 km
4. Turn **left** to merge onto **I-5 N** toward **Los Angeles** 123 km
5. Keep **left** at the fork to stay on **I-5 N**, follow signs for **Santa Ana** 63.4 km
6. Keep **right** at the fork to stay on **I-5 N**, follow signs for **I-10 W/Santa Monica/Interstate 5 N/Sacramento** 0.4 km
7. Keep **left** to continue on **I-5 N/Golden State Fwy**, follow signs for **Sacramento/Interstate 5 N** 3.3 km
8. Keep **left** to continue on **I-5 N** 33.2 km
9. Keep **left** to stay on **I-5 N** 7.7 km
10. Keep **left** at the fork to stay on **I-5 N** 2.5 km
11. Keep **left** to stay on **I-5 N** 93.9 km

Figure 5.16: Mission Direction



Start: San Diego **End:** San Diego **Travel:** Driving

Figure 5.17: Edited Mission Colors

Finally, students can customize the color of the mission city markers by changing the hex color code in between the single quotation marks on lines 102 and 105 of the Javascript code displayed in Listing 5.16. The edited Javascript code in Listing 5.17 produces the map in Figure 5.17 by changing the mission city color from red to purple. The hex color code on line 105 changes the color that fills the circle and the hex color code on line 102 changes the color of the border around the circle.

Listing 5.16: Original Javascript Code for Mission City Colors

```
99  for (var city in citymap) {
100     var cityCircle = new google.maps.Circle({
101         /* CHANGE COLOR OF THE CITIES HERE */
102         strokeColor: '#FF0000',
103         strokeOpacity: 0.8,
104         strokeWeight: 1,
105         fillColor: '#FF0000',
106         fillOpacity: 0.5,
107         map: map,
108         center: citymap[city].center,
109         radius: 8000
110     });
111 }
```

Listing 5.17: **Edited Javascript Code to Change Mission City Colors**

```
for (var city in citymap) {
    var cityCircle = new google.maps.Circle({
        /* CHANGE COLOR OF THE CITIES HERE */
        strokeColor: '#660099',
        strokeOpacity: 0.8,
        strokeWeight: 1,
        fillColor: '#d580ff',
        fillOpacity: 0.5,
        map: map,
        center: citymap[city].center,
        radius: 8000
    });
}
```

5.4 Session Discussion

Approximately 5 minutes are reserved at the end of each session in order to conclude the lesson. The time is dedicated for follow up questions and discussions. The discussions depend on the standards for each grade level. Students can also ask questions about the lesson that they are curious about during this time.

5.4.1 Discussion: Geography

The list below outlines the questions that are asked during the last few minutes of the Geography session based on the standards in Table 5.5.

For a 4th grade classroom:

- What is the estimated population of California to the nearest hundred thousand?

- What is the estimated population of New York to the nearest million?

For a 5th grade classroom:

- What are the two states with the highest populations?
- What are the two states with the lowest populations?

For a 6th grade classroom:

- What is the ratio of California population to Mississippi population?
- What is the ratio of states with population above 20 million to states with population below 20 million?

Table 5.5: Standards Covered by Geography Lesson

Grade	Standard	Description
4th	CCSS.MATH.CONTENT 4.NBT.A.3	Use place value understanding to round multi-digit whole numbers to any place.
5th	HSS 5.9	Know the location of the current 50 states and the names of their capitals.
6th	CCSS.MATH.CONTENT 6.RP.A.1	Understand the concept of a ratio and use ratio language to describe a ratio relationship between two quantities.

5.4.2 Discussion: Writing

The standards for this session, displayed in Table 5.6, are demonstrated within the lab activity itself, but a discussion is still held in order to conclude the session. Students have the chance during this time to share what they wrote and what they would like to make a website about if they were to do this activity again.

Table 5.6: Standards Covered by Writing Lesson

Grade	Standard	Description
4th	CCSS.ELA-LITERACY W.4.2	Write informative/explanatory texts to examine a topic and convey ideas and information clearly.
5th	CCSS.ELA-LITERACY W.5.2	Write informative/explanatory texts to examine a topic and convey ideas and information clearly.
6th	CCSS.ELA-LITERACY W.6.4	Produce clear and coherent writing in which the development, organization, and style are appropriate to task, purpose, and audience.

5.4.3 Discussion: Graphs

Table 5.7 displays the standards that the wrap up discussion focuses on.

For a 4th grade classroom:

- Which day of the month has the highest temperature?
- Which day of the month has the lowest temperature?

For a 5th grade classroom:

- Which two days of the month have the same temperature?
- Which day of the month is the hottest day?

For a 6th grade classroom:

- Out of 100 hours, how many hours did you spend sleeping?
- What is the percentage of the time you spent eating and doing homework?

Table 5.7: Standards Covered by Graphs Lesson

Grade	Standard	Description
4th	NGSS.SCIENCE 4.6	Construct and interpret graphs from measurements.
5th	CCSS.MATH.CONTENT 5.G.A.2	Represent real world and mathematical problems by graphing points in the first quadrant of the coordinate plane, and interpret coordinate values of points in the context of the situation.
6th	CCSS.MATH.CONTENT 6.RP.A.3.C	Find a percent of a quantity as a rate per 100; solve problems involving finding the whole, given a part and the percent.

5.4.4 Discussion: California Missions

This section lists the questions that are asked at the end of the California Missions session based on the standards in Table 5.8.

For a 4th grade classroom:

- Looking at the missions on the map, can you name the missions in the order by the year they were established?
- Looking at the missions on the map, can you tell if their locations have anything in common?

For a 5th grade classroom:

- Find the distance between any two missions. How would you convert that distance into meters? What is the distance in meters?

For a 6th grade classroom:

- Find the distance and travel time by driving between any two missions. At what rate would you be driving?

Table 5.8: Standards Covered by California Missions Lesson

Grade	Standard	Description
4th	HSS 4.2.3	Describe the Spanish exploration and colonization of California, including the relationships among soldiers, missionaries, and Indians (e.g., Juan Crespi, Junipero Serra, Gaspar de Portola).
5th	CCSS.MATH.CONTENT 5.MD.A.1	Convert among different-sized standard measurement units within a given measurement system (e.g., convert 5 cm to 0.05 m), and use these conversions in solving multi-step, real world problems.
6th	CCSS.MATH.CONTENT 6.RP.A.2	Understand the concept of a unit rate a/b associated with a ratio a:b with b $\neq$ 0, and use rate language in the context of a ratio relationship.

Chapter 6

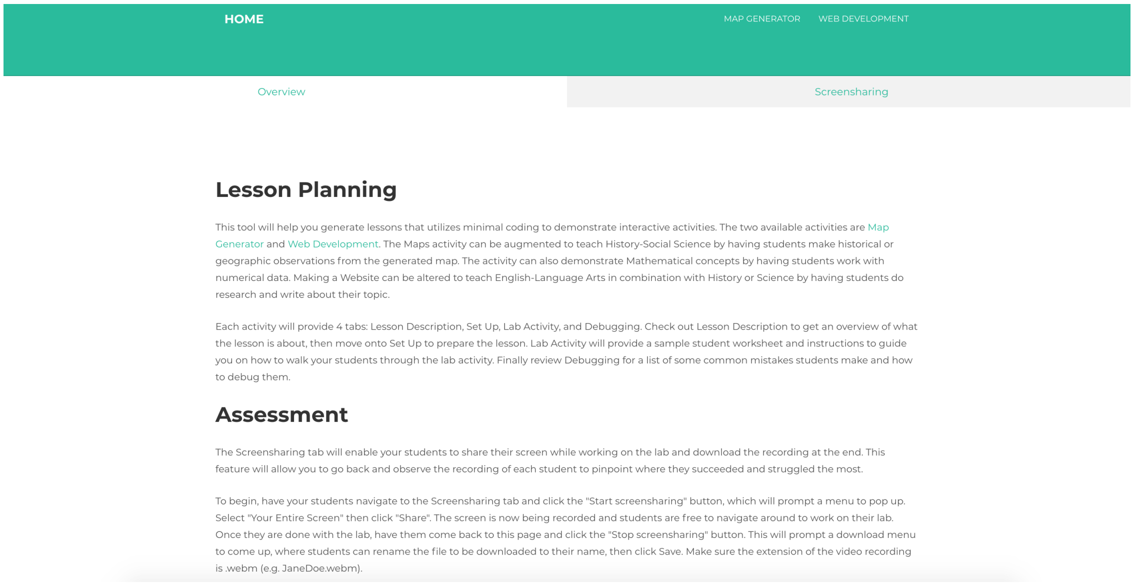
TEACHING TOOL

From my interactions with several primary school teachers, they are supportive of teaching their students about computer science but do not have the technical background to be the one teaching those lessons. The web-based teaching tool developed in this thesis work aims to help teachers feel more comfortable and confident integrating computer science into their classroom. The tool provides detailed instructions for teachers to set up and teach two of the lessons from the Integrated Computer Science Unit (Geography and Writing) and a screensharing functionality for teachers to assess where students did well and where they struggled. The Geography lesson was chosen because the teachers expressed the most interest in the lesson idea, and the Writing lesson was chosen because it is the most flexible to integrate a variety of subjects by requiring students to write about any subject the teacher would prefer.

6.1 Design

This web-based tool has three pages: Home, Map Generator, and Web Development. All instructions on the website are described in detail and accompanied by images to help teachers feel confident that they are following the steps correctly.

The Home page, shown in Figure 6.1, is the front page that teachers will land on when they navigate to the website. The page has two tabs - Overview and Screensharing. The Overview tab explains how teachers can navigate through the website to generate lessons and provides directions for using the screensharing feature. The Screensharing tab, shown in Figure 6.2, allows the tool to record and save the students' screen to capture everything they are doing during the lab. Teachers can go back to these recordings to observe where students struggled the most and adjust teaching strategies accordingly.



Lesson Planning

This tool will help you generate lessons that utilizes minimal coding to demonstrate interactive activities. The two available activities are Map Generator and Web Development. The Maps activity can be augmented to teach History-Social Science by having students make historical or geographic observations from the generated map. The activity can also demonstrate Mathematical concepts by having students work with numerical data. Making a Website can be altered to teach English-Language Arts in combination with History or Science by having students do research and write about their topic.

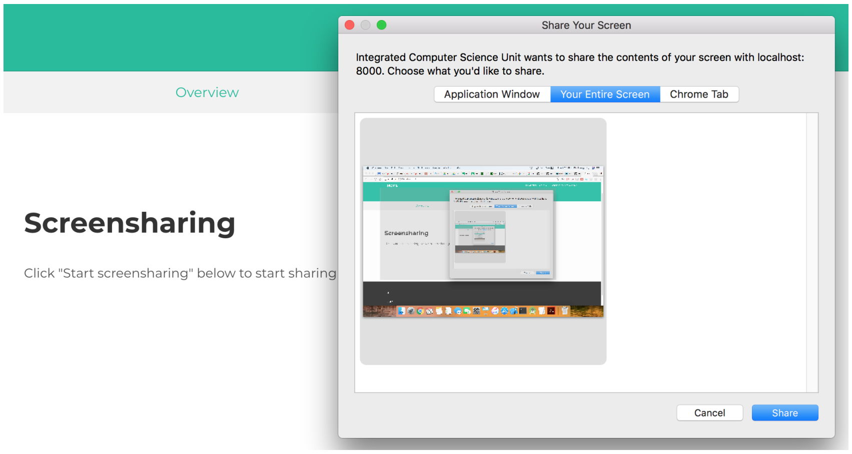
Each activity will provide 4 tabs: Lesson Description, Set Up, Lab Activity, and Debugging. Check out Lesson Description to get an overview of what the lesson is about, then move onto Set Up to prepare the lesson. Lab Activity will provide a sample student worksheet and instructions to guide you on how to walk your students through the lab activity. Finally review Debugging for a list of some common mistakes students make and how to debug them.

Assessment

The Screensharing tab will enable your students to share their screen while working on the lab and download the recording at the end. This feature will allow you to go back and observe the recording of each student to pinpoint where they succeeded and struggled the most.

To begin, have your students navigate to the Screensharing tab and click the "Start screensharing" button, which will prompt a menu to pop up. Select "Your Entire Screen" then click "Share". The screen is now being recorded and students are free to navigate around to work on their lab. Once they are done with the lab, have them come back to this page and click the "Stop screensharing" button. This will prompt a download menu to come up, where students can rename the file to be downloaded to their name, then click Save. Make sure the extension of the video recording is .webm (e.g. JaneDoe.webm).

Figure 6.1: Home Page



Screensharing

Click "Start screensharing" below to start sharing

Integrated Computer Science Unit wants to share the contents of your screen with localhost: 8000. Choose what you'd like to share.

Figure 6.2: Screensharing Tab

The Map Generator page allows teachers to create a lesson by choosing from different maps, including all of the U.S. states and world continents, and several different countries. A Google spreadsheet is linked into the source code, where the values in the spreadsheet determine how the map is populated. The purpose of the instructions on this page is to help teachers set up and replicate the Geography lesson from the Integrated Computer Science Unit.

The Web Development page guides teachers to develop lessons focused on writing. Teachers are allowed the flexibility of integrating any subject with this computer science lesson that they want students to write about. The purpose of the instructions on this page is to help teachers replicate the Writing lesson from the Integrated Computer Science Unit.

There are two tabs under the Home page and four tabs under each lesson - Lesson Overview, Set Up, Lab Activity, and Debugging, as shown in Figure 6.3. The rest of this section discusses the content of each tab.

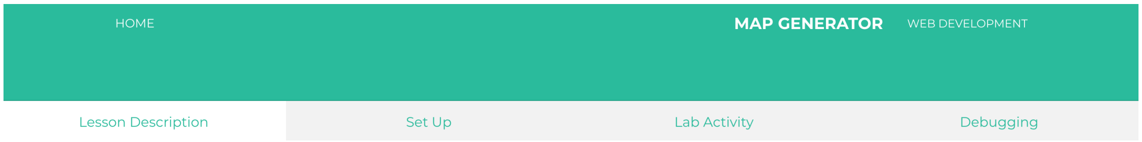


Figure 6.3: Tabs under each lesson

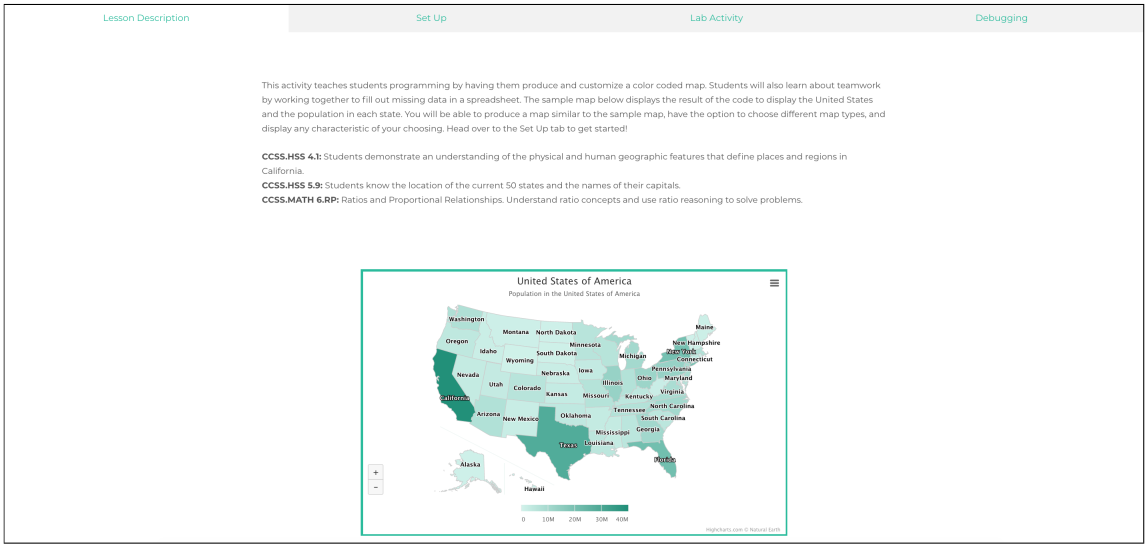
6.1.1 Lesson Description

This tab provides a summary of the lesson, displays a sample of the end product, and includes a set of possible standards that can be integrated into this computer science activity. The lesson description for the Map Generator lesson is displayed in Figure 6.5 and the lesson description for the Web Development lesson in Figure 6.4.

6.1.2 Set Up

Map Generator

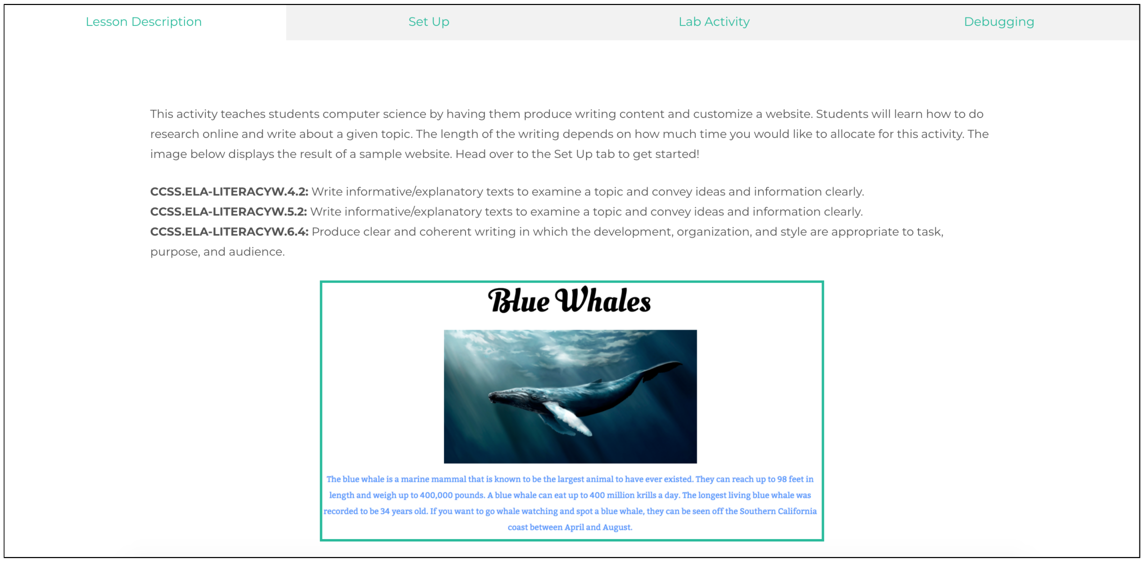
The Set Up tab, displayed in Figure 6.6, provides a walk-through to help teachers set up the map for the Geography lesson. First, the teacher is instructed on how to



This activity teaches students programming by having them produce and customize a color coded map. Students will also learn about teamwork by working together to fill out missing data in a spreadsheet. The sample map below displays the result of the code to display the United States and the population in each state. You will be able to produce a map similar to the sample map, have the option to choose different map types, and display any characteristic of your choosing. Head over to the Set Up tab to get started!

CCSS.HSS 4.1: Students demonstrate an understanding of the physical and human geographic features that define places and regions in California.
CCSS.HSS 5.9: Students know the location of the current 50 states and the names of their capitals.
CCSS.MATH 6.RP: Ratios and Proportional Relationships. Understand ratio concepts and use ratio reasoning to solve problems.

Figure 6.4: Map Generator: Lesson Description



This activity teaches students computer science by having them produce writing content and customize a website. Students will learn how to do research online and write about a given topic. The length of the writing depends on how much time you would like to allocate for this activity. The image below displays the result of a sample website. Head over to the Set Up tab to get started!

CCSS.ELA-LITERACYW.4.2: Write informative/explanatory texts to examine a topic and convey ideas and information clearly.
CCSS.ELA-LITERACYW.5.2: Write informative/explanatory texts to examine a topic and convey ideas and information clearly.
CCSS.ELA-LITERACYW.6.4: Produce clear and coherent writing in which the development, organization, and style are appropriate to task, purpose, and audience.

Blue Whales

The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day. The longest living blue whale was recorded to be 34 years old. If you want to go whale watching and spot a blue whale, they can be seen off the Southern California coast between April and August.

Figure 6.5: Web Development: Lesson Description

publish their Google spreadsheet so that JSFiddle can access the data and how they can input data into the spreadsheet in the correct format. Next, teachers can choose from a dropdown list of what type of map they would like to work with, which will generate the next set of instructions based on that selection. When the teacher opens the provided JSFiddle link, most of the source code will already be prepared. All the teacher has to do is paste the key value of the map type they chose and the key of their Google spreadsheet into the provided source code. The original code snippets are shown in Listings 6.1 and 6.2 and the expected code snippets to finish setting up are shown in Listings 6.3 and 6.4.

Listing 6.1: Original HTML Code

```html
<script src="https://code.highcharts.com/mapdata/PASTE MAP TYPE
   HERE.js"></script>
```

Listing 6.2: Original Javascript Code

```javascript
$(function () {
    Highcharts.data({
        googleSpreadsheetKey: 'PASTE GOOGLE SPREADSHEET KEY HERE',
        parsed: function (columns) {
```

Listing 6.3: Expected HTML Code for a United States map

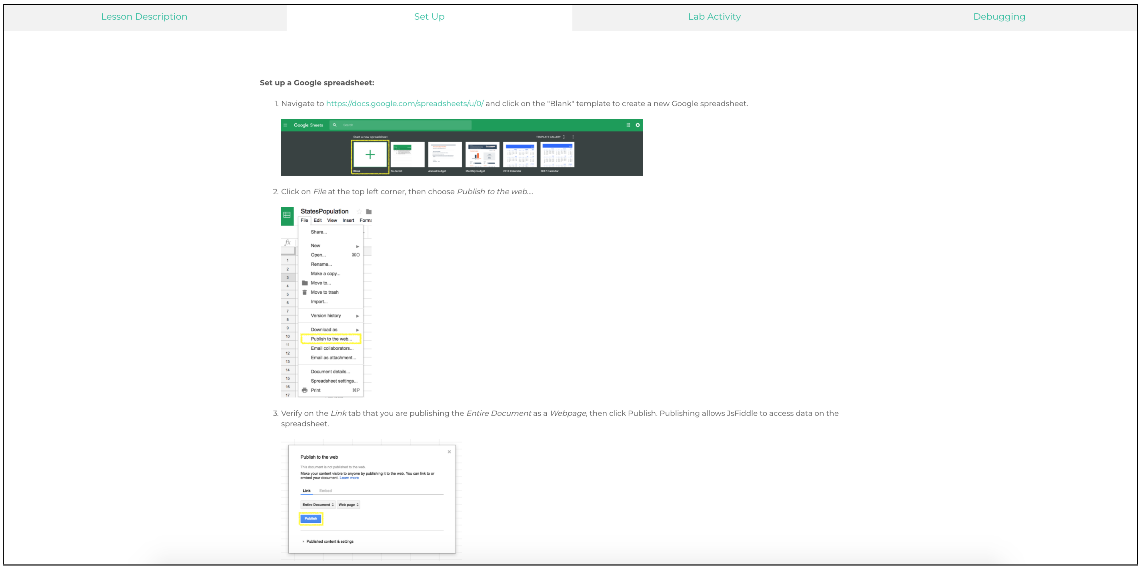
```html
<script src="https://code.highcharts.com/mapdata/countries/us/
   us-all.js"></script>
```

Listing 6.4: Expected Javascript Code for Google Spreadsheet

```javascript
$(function () {
    Highcharts.data({
        googleSpreadsheetKey: '1
           S32WV8GWCTwaMhELWXWPw4nY_kdA6J2rXwRXJkhLU4M',
        parsed: function (columns) {
```

Web Development

Displayed in Figure 6.7 is the Set Up tab for the lesson. Teachers are instructed to navigate to the CodePen editor to confirm that they see the expected code and loaded website in the output panel.



Set up a Google spreadsheet:

1. Navigate to https://docs.google.com/spreadsheets/u/0/ and click on the "Blank" template to create a new Google spreadsheet.

2. Click on *File* at the top left corner, then choose *Publish to the web....*

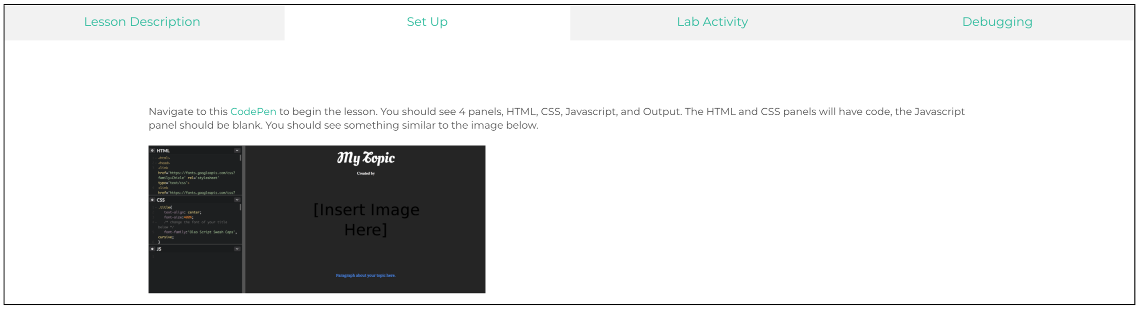
3. Verify on the *Link* tab that you are publishing the *Entire Document* as a *Webpage*, then click Publish. Publishing allows JsFiddle to access data on the spreadsheet.

Publish to the web

This document is not published to the web.

Make your content visible to anyone by publishing it to the web. You can link to or embed your document. Learn more

Figure 6.6: Map Generator: Set Up



Navigate to this CodePen to begin the lesson. You should see 4 panels, HTML, CSS, Javascript, and Output. The HTML and CSS panels will have code, the Javascript panel should be blank. You should see something similar to the image below.

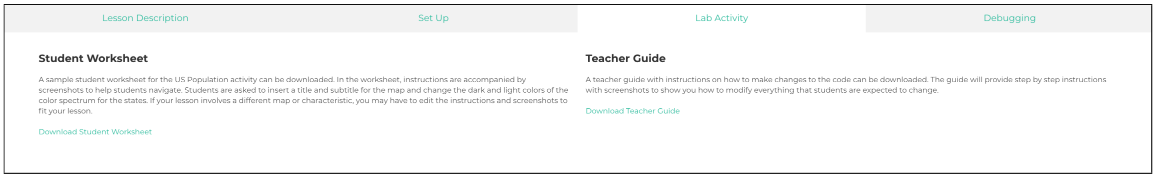
```
<html>
<head>
<link
href='https://fonts.googleapis.com/css?
family=Chicle' rel='stylesheet'
type='text/css'>
<link
href='https://fonts.googleapis.com/css?
```

```
.title{
    text-align: center;
    font-size:400%;
    /* change the font of your title
below */
    font-family:'Oleo Script Swash Caps',
cursive;
}
```

Figure 6.7: Web Development: Set Up

6.1.3 Lab Activity

The lab activity tab under each lesson includes two downloadable documents, one is an activity worksheet for the students and one is a guide for the teacher. The worksheet lays out step-by-step instructions for what the students are expected to do during lab, such as finding the correct line numbers to change the text and colors. Teachers can choose to give the worksheet to students to complete on their own or use it to help themselves with the demo while having students follow along for each step. The teacher guide provides more in depth explanation on how to make changes to the code in order to help them explain the procedure to their students during lecture. The lab activity tab for the Map Generator and Web Development lessons are shown in Figure 6.8 and Figure 6.9 respectively. The student worksheets and teacher guides are provided in Appendix A.



Student Worksheet

A sample student worksheet for the US Population activity can be downloaded. In the worksheet, instructions are accompanied by screenshots to help students navigate. Students are asked to insert a title and subtitle for the map and change the dark and light colors of the color spectrum for the states. If your lesson involves a different map or characteristic, you may have to edit the instructions and screenshots to fit your lesson.

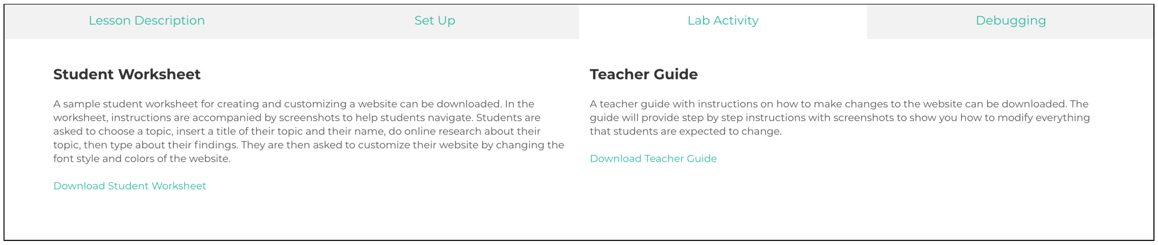
Download Student Worksheet

Teacher Guide

A teacher guide with instructions on how to make changes to the code can be downloaded. The guide will provide step by step instructions with screenshots to show you how to modify everything that students are expected to change.

Download Teacher Guide

Figure 6.8: Map Generator: Lab Activity



Student Worksheet

A sample student worksheet for creating and customizing a website can be downloaded. In the worksheet, instructions are accompanied by screenshots to help students navigate. Students are asked to choose a topic, insert a title of their topic and their name, do online research about their topic, then type about their findings. They are then asked to customize their website by changing the font style and colors of the website.

Download Student Worksheet

Teacher Guide

A teacher guide with instructions on how to make changes to the website can be downloaded. The guide will provide step by step instructions with screenshots to show you how to modify everything that students are expected to change.

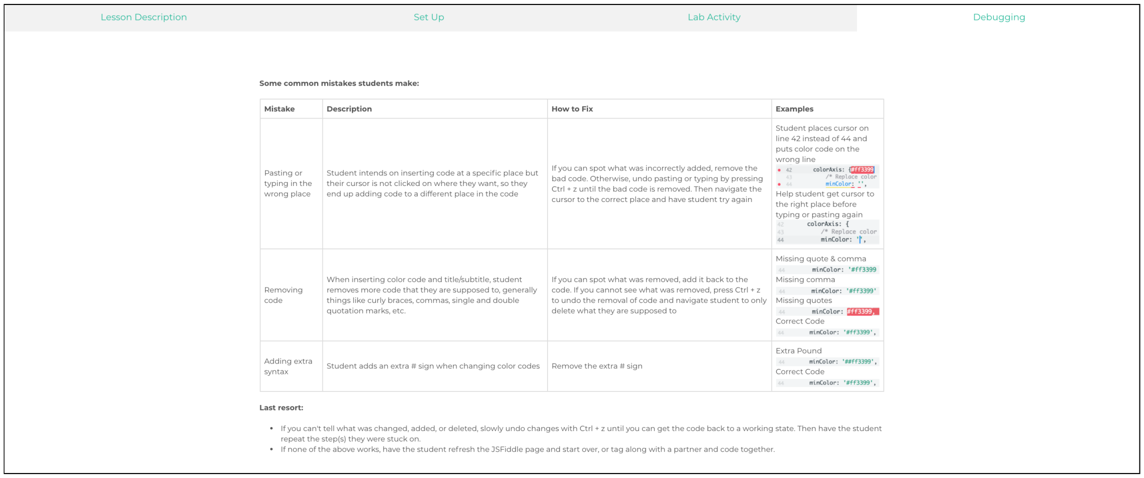
Download Teacher Guide

Figure 6.9: Web Development: Lab Activity

6.1.4 Debugging

Based on the observations made in the four computer science sessions, common mistakes are compiled and solutions are suggested on this tab. Figure 6.10 and Figure 6.11 display the debugging suggestions for the Map Generator and Web Development lessons respectively.

1. **Pasting or typing in code at the wrong location**. Students would intend to paste the hex color code at a specific location, but end up pasting elsewhere because their cursor is not clicked where they thought. Listing 6.5 shows an example of this mistake.



Some common mistakes students make:

Mistake	Description	How to Fix	Examples

Pasting or typing in the wrong place	Student intends on inserting code at a specific place but their cursor is not clicked on where they want, so they end up adding code to a different place in the code	If you can spot what was incorrectly added, remove the bad code. Otherwise, undo pasting or typing by pressing Ctrl + z until the bad code is removed. Then navigate the cursor to the correct place and have student try again	Student places cursor on line 42 instead of 44 and puts color code on the wrong line
			`42 colorAxis: {#ff3399`
			`43 /* Replace color`
			`44 minColor: '',`
			Help student get cursor to the right place before typing or pasting again
			`42 colorAxis: {`
			`43 /* Replace color`
			`44 minColor: '',`

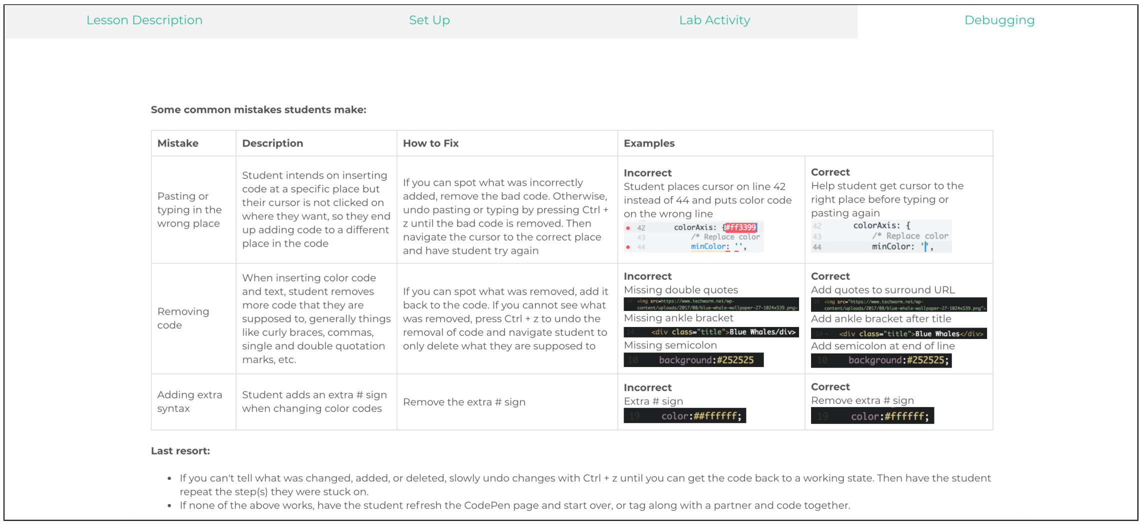
Removing code	When inserting color code and title/subtitle, student removes more code that they are supposed to, generally things like curly braces, commas, single and double quotation marks, etc.	If you can spot what was removed, add it back to the code. If you cannot see what was removed, press Ctrl + z to undo the removal of code and navigate student to only delete what they are supposed to	Missing quote & comma
			`44 minColor: '#ff3399`
			Missing comma
			`44 minColor: '#ff3399'`
			Missing quotes
			`44 minColor: #ff3399,`
			Correct Code
			`44 minColor: '#ff3399',`

Adding extra syntax	Student adds an extra # sign when changing color codes	Remove the extra # sign	Extra Pound
			`44 minColor: '##ff3399',`
			Correct Code
			`44 minColor: '#ff3399',`

Last resort:

- If you can't tell what was changed, added, or deleted, slowly undo changes with Ctrl + z until you can get the code back to a working state. Then have the student repeat the step(s) they were stuck on.
- If none of the above works, have the student refresh the JSFiddle page and start over, or tag along with a partner and code together.

Figure 6.10: Map Generator: Debugging



Some common mistakes students make:

Mistake	Description	How to Fix	Examples	

Pasting or typing in the wrong place	Student intends on inserting code at a specific place but their cursor is not clicked on where they want, so they end up adding code to a different place in the code	If you can spot what was incorrectly added, remove the bad code. Otherwise, undo pasting or typing by pressing Ctrl + z until the bad code is removed. Then navigate the cursor to the correct place and have student try again	**Incorrect**
			Student places cursor on line 42 instead of 44 and puts color code on the wrong line
			`42 colorAxis: {#ff3399`
			`43 /* Replace color`
			`44 minColor: '',`
				Correct
				Help student get cursor to the right place before typing or pasting again
				`42 colorAxis: {`
				`43 /* Replace color`
				`44 minColor: '',`

Removing code	When inserting color code and text, student removes more code that they are supposed to, generally things like curly braces, commas, single and double quotation marks, etc.	If you can spot what was removed, add it back to the code. If you cannot see what was removed, press Ctrl + z to undo the removal of code and navigate student to only delete what they are supposed to	**Incorrect**
			Missing double quotes
			`<img src=https://www.techworm.net/wp-content/uploads/2017/08/blue-whale-wallpaper-27-1024x539.png>`
			Missing ankle bracket
			`<div class="title">Blue Whales/div>`
			Missing semicolon
			`background:#252525`
				Correct
				Add quotes to surround URL
				`<img src="https://www.techworm.net/wp-content/uploads/2017/08/blue-whale-wallpaper-27-1024x539.png">`
				Add ankle bracket after title
				`<div class="title">Blue Whales</div>`
				Add semicolon at end of line
				`background:#252525;`

Adding extra syntax	Student adds an extra # sign when changing color codes	Remove the extra # sign	**Incorrect** Extra # sign `color:##ffffff;`	**Correct** Remove extra # sign `color:#ffffff;`

Last resort:

- If you can't tell what was changed, added, or deleted, slowly undo changes with Ctrl + z until you can get the code back to a working state. Then have the student repeat the step(s) they were stuck on.
- If none of the above works, have the student refresh the CodePen page and start over, or tag along with a partner and code together.

Figure 6.11: Web Development: Debugging

2. **Getting rid of syntax they were not supposed to (e.g. (a) surrounding quotation marks and (b) comma when replacing hex color codes)**. Students would intend to highlight the hex color code in order to replace it with a new color code, but they end up highlighting more than needed and remove the required syntax. Listing 6.6 shows an example of this mistake.

3. **Adding extra syntax**. This is the most straightforward mistake and easiest to spot. Students simply do not remove the original pound sign along with the

hex color code but paste in the pound sign along with their new color code, thus ending up with two pound signs instead of one. Listing 6.7 displays an example of this mistake.

Listing 6.5: Map Mistake (1)

```
colorAxis: {#ff3399
    minColor: '',
}
```

Listing 6.6: Map Mistake (2)

```
colorAxis: {
    minColor: '#ff3399
}
```

Listing 6.7: Map Mistake (3)

```
colorAxis: {
    minColor: '##ff3399',
}
```

Listing 6.8: Map Correct Code

```
colorAxis: {
    minColor: '#ff3399',
}
```

The correct code for the mentioned mistakes is shown is Listing 6.8. Although the examples only demonstrate the mistakes made with a hex color code, the same mistakes and corrections can be applied when inserting text.

Chapter 7

VALIDATION

7.1 Integrated Computer Science Unit

This section discusses the research approval, survey, and results of the computer science lessons. The purpose of this study is to expose students to computer science and obtain results on whether the students have demonstrated better knowledge of and expressed increased interest in Computer Science after the sessions.

7.1.1 Research Approval

Since the study in this thesis involves interacting with and surveying underage participants, all materials involved in the study were reviewed and approved by the Cal Poly Institutional Review Board (IRB). Materials include surveys to gauge the students' knowledge of and interest in computer science, a Child Assent Script to receive agreement to participate from the students, a Parental Permission Form to obtain the parent or guardian approval for their child to engage in the study, a Principal Authorization for each participating elementary school, and a Human Subjects Protocol Approval Form to describe the details of the study to be conducted.

7.1.2 Survey

Two surveys are given to the students that received parental permission to participate, a pre-survey before the first session and a post-survey after the last session.

The first few questions of the pre-survey ask basic demographic information including grade level and name of school. The next few questions are meant to gauge students' interests, such as favorite subject(s) in school, career choice for the future, and students' knowledge of computer science by looking at a list of computer science related vocabulary and checking the ones they know the definition for. The remaining questions consist of attitudinal questions, such as "Do you want to go to college?" and "What do you think it takes to be an engineer?"

The post-survey contains the same set of questions with a few additional questions involving student interest in the four lessons in the computer science unit. Students are asked to choose their favorite session and explain why they favored that specific lesson the most. Students are also asked to rate how much they enjoyed the lessons based on a scale from 1 to 10 and whether they would like these lessons to continue in their classroom.

The surveys are hosted on SurveyMonkey and are given to all of the students of each classroom at the same time. The parental permission form and both surveys are provided in Appendix B.

Grover Heights Elementary School

Out of the 26 students in the 4th grade classroom at Grover Heights Elementary School, 24 students had their parental permission form signed. Out of those 24 students, 21 students took both the pre-survey and post-survey. The final results are based on the data collected from those 21 students.

Lange Elementary School

Of the 32 students in the 5th grade classroom at Lange Elementary School, 29 students had their parental permission form signed. Of those 29, 27 students filled out both the pre-survey and post-survey. Thus the data for Lange Elementary School consists of those 27 students.

Pacheco Elementary School

Of the 18 students that attended Code Club, 13 students had their permission form signed. All of those students filled out both the pre-survey and post-survey. 11 of the students are in 4th grade and 2 of the students are in 5th grade. The following results represent those 13 students.

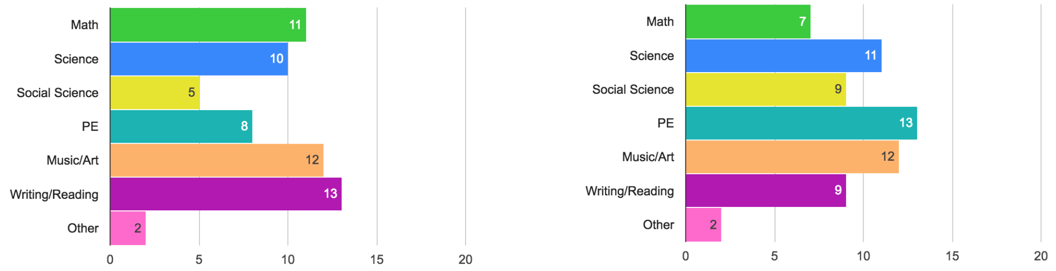
7.1.3 Results

Student Favorite Subjects in School

The results for student favorite subjects do not seem to have any correlation with the computer science lessons. Of the four subjects that were integrated in the computer science lessons (Math, Science, Social Science, and Writing/Reading), there is a slight increase in interest in Science at Grover Heights (Figure 7.1), Science and Writing/Reading at Lange (Figure 7.2), and Math, Social Science, and Writing/Reading at Pacheco (Figure 7.3). The computer science lessons did not cover PE and Music/Art, but the data indicate that students' interest in PE and Music/Art have also changed. The fluctuation in the students' favorite subjects is most likely due to their experience during normal school hours and it is not clear whether the computer science lessons played a part. These results suggest that students of this age range are open to different interests and are likely to change their mind on a daily and weekly basis. It will require more than teaching computer science for an hour a week to observe students' long term interests in computer science.

Student Computer Science Vocabulary Knowledge

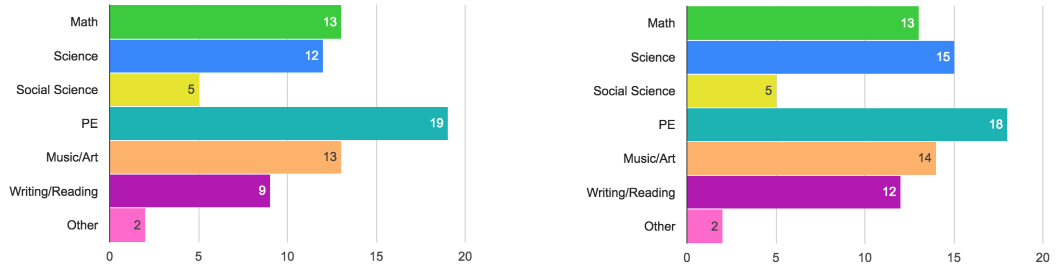
To analyze this question, each computer science vocabulary term is worth 1 point and students are assigned a score between 0 and 6 depending on how many vocabulary



(a) Pre-Survey

(b) Post-Survey

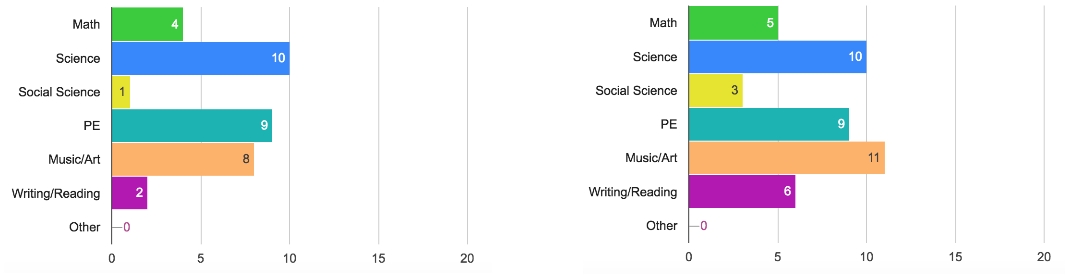
Figure 7.1: Grover Heights: Student Favorite Subjects



(a) Pre-Survey

(b) Post-Survey

Figure 7.2: Lange: Student Favorite Subjects



(a) Pre-Survey

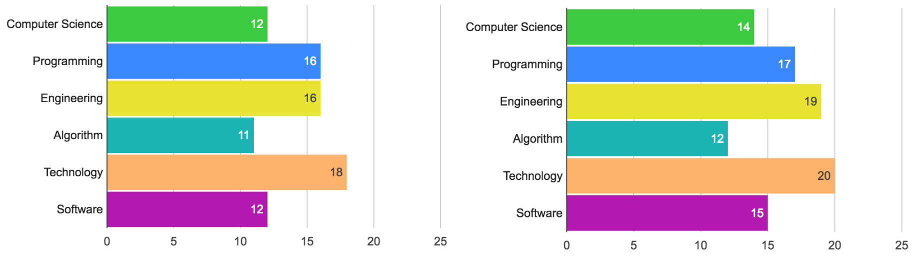
(b) Post-Survey

Figure 7.3: Pacheco: Student Favorite Subjects

terms they knew. Figures 7.4, 7.5, and 7.6 show the pre-survey and post-survey results for Grover Heights, Lange, and Pacheco respectively.

Looking only at the pre-survey, Grover Heights students averaged at 3.285, Lange students averaged at 2.778, and Pacheco students averaged at 4.153. The results indicate that Pacheco students knew the most computer science vocabulary. This is not surprising because the students at Pacheco have attended after school coding programs provided by the school in the past. Between Grover Heights and Lange, students at Grover Heights knew more computer science vocabulary than students at Lange. This is likely due to the fact that Lange has about twice the number of English learners than Grover Heights does. Demographic statistics from the schools show that Lange also has more underrepresented students and more low income students than Grover Heights, but we cannot conclude any causal relationships.

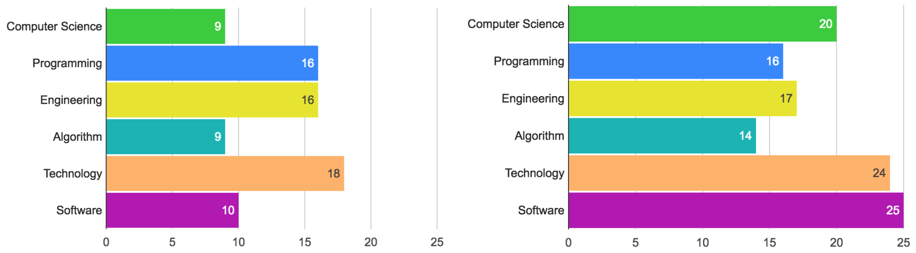
There is a statistically significant increase in computer science vocabulary knowledge from the pre-survey to the post-survey across all three schools. A paired t-test for the increase in computer science knowledge for each school yielded a p-value of 0.00042 for Grover Heights, 0.0000547 for Lange, and 0.023 for Pacheco. The average increase in points is 1.238 for Grover Heights, 1.444 for Lange, and 1.230 for Pacheco. Although the average increase in knowledge at Lange is slightly higher than the average increase in knowledge at the other two schools, the difference in means between the three schools is not statistically significant (p-value = 0.88 using the ANOVA test). The above results mean that the increase in computer science knowledge among the students is statistically meaningful but no school significantly outperformed the others.



(a) Pre-Survey

(b) Post-Survey

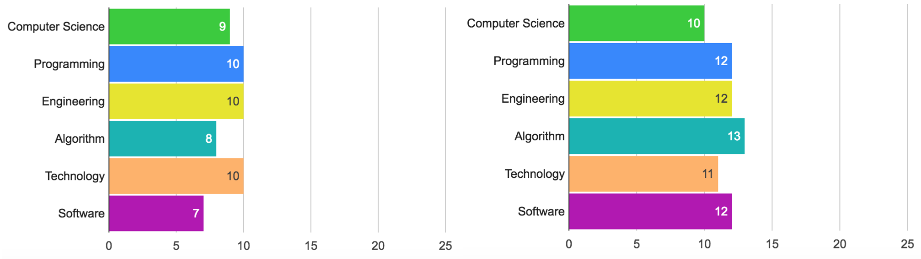
Figure 7.4: Grover Heights: Student CS Vocab Knowledge



(a) Pre-Survey

(b) Post-Survey

Figure 7.5: Lange: Student CS Vocab Knowledge



(a) Pre-Survey

(b) Post-Survey

Figure 7.6: Pacheco: Student CS Vocab Knowledge

Students Considering College

The number of students considering going to college in the future remained the same from the pre-survey to the post-survey for all three schools, as shown in Tables 7.1, 7.2, and 7.3. This is plausible because this is a more permanent decision for students than their favorite subject(s) in school, therefore it is not something students will change their mind about in a short period of time. The results in this question do not provide any indications whether the computer science lessons had an effect on the students' intention on attending college in the future. This question could provide more insight if we were to ask the students for their intended major instead.

Table 7.1: Grover Heights: Number of Students Considering College

(a) Pre-Survey

Yes	**No**	**Maybe**
17	1	4

(b) Post-Survey

Yes	**No**	**Maybe**
17	1	4

Table 7.2: Lange: Number of Students Considering College

(a) Pre-Survey

Yes	**No**	**Maybe**
22	1	4

(b) Post-Survey

Yes	**No**	**Maybe**
22	1	4

Table 7.3: Pacheco: Number of Students Considering College

(a) Pre-Survey

Yes	**No**	**Maybe**
12	1	0

(b) Post-Survey

Yes	**No**	**Maybe**
12	1	0

Student Future Job

As expected, the jobs that students want to pursue when they grow up varied for all schools, shown in Tables 7.4, 7.5, and 7.6. Although most students gave the same answer for both the pre-survey and post-survey, a few students did change their answer. One student from Grover Heights who initially did not know what future career they wanted to pursue and one student from Lange who initially wanted to work at a pizza shop both changed their answer and indicated that they want to become a programmer/coder when they grow up.

It is expected that introducing computer science to a group of students will not spark interest for everyone and not all students will become interested in computer science, but it is the hope that all students will have equal exposure to computer science and know that it is a career option.

Table 7.4: Grover Heights: Student Future Job

Pre-Survey: # of Students	**Post Survey: # of Students**	**Field**	**Student Answers**
3	2	Sports	NFL, Gymnast
4	5	Entertainment	Actress, Singer, YouTuber
3	4	Engineering/ Technology	Game Designer, IT, **Coder**
3	3	Science/Math	Astronomer, Doctor
2	2	Public Service	Firefighter, Police
2	4	Education	Teacher
3	0	Other	Zoo Keeper, Event Planner
2	2	Don't Know	Don't Know/Blank

Table 7.5: Lange: Student Future Job

Pre-Survey: **# of Students**	**Post Survey:** **# of Students**	**Field**	**Student Answers**
5	5	Sports	MLB, NBA, Rally Racing
3	3	Entertainment	Actor, Photographer
2	2	Engineering/ Technology	Architect, Civil Engineer, **Programmer**
6	7	Science/Math	Anesthesiologist, Doctor
4	4	Public Service	Police, Detective, Army
2	2	Education	Teacher
2	3	Other	Pizza Shop, Zoo Keeper
3	1	Don't Know	Don't Know/Blank

Table 7.6: Pacheco: Student Future Job

Pre-Survey: **# of Students**	**Post Survey:** **# of Students**	**Field**	**Student Answers**
2	2	Sports	NBA, Soccer
0	0	Entertainment	None
1	1	Engineering/ Technology	Engineer
3	4	Science/Math	Marine Biologist, Chemist
2	2	Public Service	Police, Army
2	2	Education	Teacher
2	1	Other	Dog Trainer, Food Critic
1	1	Don't Know	Don't Know/Blank

Students Considering Working with Technology

The number of students selecting each of the *Yes*, *No*, and *Maybe* options when asked if they intend to work with technology in the future fluctuated slightly in each of the school. Most students indicated the same answer from the pre-survey to the post-survey. Several students did change their answers, but there is no statistical significance in the changes. In other words, it is most likely that the changes in students' answers occurred by chance.

Similar to the previous questions about attending college and desired future job, this question is also long-term. Perhaps students are not yet able to make a decision for this question from experiencing only a few computer science lessons.

Table 7.7: Grover Heights: Number of Students Considering Working with Technology in the Future

(a) Pre-Survey

Yes	**No**	**Maybe**
4	7	11

(b) Post-Survey

Yes	**No**	**Maybe**
6	5	11

Table 7.8: Lange: Number of Students Considering Working with Technology in the Future

(a) Pre-Survey

Yes	**No**	**Maybe**
6	9	12

(b) Post-Survey

Yes	**No**	**Maybe**
8	11	7

What Students Think it Takes to be an Engineer

This question is open-ended with the goal to observe if there would be any changes in students' opinion in what they think it takes to be an engineer.

Table 7.9: Pacheco: Number of Students Considering Working with Technology in the Future

(a) Pre-Survey

Yes	**No**	**Maybe**
6	2	5

(b) Post-Survey

Yes	**No**	**Maybe**
4	1	8

The results for Grover Heights and Pacheco appear to be inconclusive since the numbers fluctuated in an unexpected way. The expectation for this question was that the number of students in each category (except the Don't Know category) would increase, since being a good engineer generally requires those attributes. It is surprising that the number of students from both of these schools decreased in the *Be Smart* category from the pre-survey to the post-survey. Perhaps the students only mentioned the attributes that they felt are more important, such as working hard and being creative, and left out the ones that they felt were less important. The results for Lange seem to match expectations more, where the number of students in each category (except the Don't Know category) either stayed the same or increased.

Despite the inconclusive results, it is interesting that students came up with similar answers all by themselves, given that this question is open-ended.

Students in Favor of More Computer Science Lessons

The number of students who want to do more computer science lessons in their class is a promising sign for integrating computer science. 77.2% of students from Grover answered *Yes*, 44.4% of students from Lange answered *Yes*, and 53.8% of students from Pacheco said *Yes* in favor of having more computer science lessons in their classroom. A large percentage of students from Lange answered *Maybe* and a smaller percentage of students answered *No* while only a small percentage of students from Grover Heights and Pacheco said *Maybe* and no student from these two schools said

Table 7.10: Grover Heights: What Students Think It Takes to be an Engineer

What It Takes	Hard Work	Be Smart	Not Give Up	Be Creative	Team Player	Don't Know
Pre-Survey: # of Students	9	9	6	2	3	3
Post-Survey: # of Students	12	4	5	5	3	0

Table 7.11: Lange: What Students Think It Takes to Be an Engineer

What It Takes	Hard Work	Be Smart	Not Give Up	Be Creative	Don't Know
Pre-Survey: # of Students	9	8	5	3	6
Post-Survey: # of Students	12	8	7	5	3

No to this question. An explanation for this could be that students from Grover Heights and Pacheco are generally a year younger than those from Lange, so they are more open to the idea of incorporating something new, such as the computer science lessons, in their classroom.

It is not too worrisome that some students answered *Maybe* or *No* in response to having more computer science lessons in their classroom, because not every student enjoys every subject in school. If we were to ask students the same question but in regards to the subjects they learn in school instead of Computer Science (e.g. "Would

Table 7.12: Pacheco: What Students Think It Takes to be an Engineer

What It Takes	Hard Work	Be Smart	Not Give Up	Be Creative	Team Player	Don't Know
Pre-Survey: # of Students	4	5	2	0	1	3
Post-Survey: # of Students	8	3	2	2	1	2

you like to do more Math lessons in your classroom?"), it is possible we may end up seeing a similar spread in *Yes*, *No*, and *Maybe* answers.

Table 7.13: Grover: Number of Students Wanting to Do More CS Lessons

Yes	No	Maybe
17	0	5

Table 7.14: Lange: Number of Students Wanting to Do More CS Lessons

Yes	No	Maybe
12	4	11

Students' Most and Least Favorite Lesson

This question asked students for one most favorite lesson and one least favorite lesson, but many students could not choose one. The students' answers included either one lesson, multiple lessons, or none.

Across all three participating schools, there is a strong preference towards the Writing lesson and the California Missions lesson. Table 7.16 shows the lesson preference for students from Grover Heights. 33.3% of students preferred the Writing

Table 7.15: Pacheco: Number of Students Wanting to Do More CS Lessons

Yes	No	Maybe
7	0	6

lesson and 47.6% preferred the California Missions lesson, while only 4% preferred the Geography lesson and 19% preferred the Graphs lesson. Table 7.17 displays the same information for Lange, where 55.5% of students favored the Writing lesson and 48.1% favored the California Missions lesson, while only 3.7% preferred the Geography lesson and 14.8% preferred the Graphs lesson. For Pacheco, shown in Table 7.18, 61.5% of students preferred the Writing lesson, 53.8% preferred the California Missions lesson, 15.3% preferred the Geography lesson, and 30.7% preferred the Graphs lesson.

Adding up students from the three participating schools, 30 students chose the Writing lesson as their favorite lesson and 30 students chose the California Missions lesson as their favorite lesson. Only 10 students chose the Writing lesson as their least favorite while 17 students chose the California Missions lesson as their least favorite.

Overall, the Writing lesson was the most popular among students from all three schools. Many students mentioned in their feedback that the Writing lesson is their favorite because they were given the most freedom. Students were allowed to choose a topic for themselves, which was not the case for the other lessons. Students were able to independently do online research to find images and information about their topic. Although this lesson required more work than the rest of the lessons, students were most eager because they were able to choose a topic that they liked.

The California Missions lesson was another popular lesson among the students, but based on classroom observations and a closer analysis of the feedback, the popularity can be attributed to extrinsic reasons. According to the students' feedback, most students favored this lesson because they were able to drag the map around, zoom in to find locations they wanted to look for, and view the map in street view. This was the most difficult lesson to teach because students were often distracted by interacting with the map and did not pay attention to the code. Interestingly, despite the California Missions lesson being one of the top two most favorite lessons, it is also the least favorite lesson according to students from Grover Heights and Lange. Most of the students who chose the California Missions lesson as their least favorite indicated that they thought the lesson was too easy and a few students indicated that they did not understand the lesson due to other students being distracting. The students that found the California Missions lesson too easy perhaps thought so because the lesson reiterated many old concepts and did not teach too many new things. It is actually a promising indicator if students found the lesson easy, because that means they have retained information from previous lessons.

The Geography lesson had the smallest number of students that indicated it was their favorite lesson. The reason for this is unclear because these students did not leave feedback explaining themselves. There are several explanations that could be plausible for this. One possibility for students at Lange is that this lesson experienced some technical difficulty that resulted in a five minute delay in the lesson, which could have caused confusion for that group of students. A possibility for students at Pacheco is that on the day of the Geography lesson, the students did not receive their normal recess and lunch time due to a rainy day. The lack of play time during school hours caused the students to be extremely energetic and distracted after school. Another possibility for all schools could be that since the Geography lesson was the first lesson that was taught, some students may have forgotten about it four weeks later.

Table 7.16: Grover Heights: Most and Least Favorite Sessions

	Geography	Writing	Graphs	California Missions
Most Favorite: # of Students	1	7	4	10
Least Favorite: # of Students	3	5	6	8

Table 7.17: Lange: Most and Least Favorite Sessions

	Geography	Writing	Graphs	California Missions
Most Favorite: # of Students	1	15	4	13
Least Favorite: # of Students	2	4	4	8

Table 7.18: Pacheco: Most and Least Favorite Sessions

	Geography	Writing	Graphs	California Missions
Most Favorite: # of Students	2	8	4	7
Least Favorite: # of Students	2	1	5	1

Student Rating of Computer Science Lessons

On a scale of 1 to 10, with 1 being the lowest and 10 being the highest, 40.9% of students at Grover Heights said *Yes* in favor of doing more computer science lessons rated the computer science lessons a 10 compared to 37% of students at Lange and 38.4% of students at Pacheco. The average rating from students at Grover is 8.28, the average rating from students at Lange is 8.18, and the average rating from students at Pacheco is 7.92. The ANOVA test for the difference in means of the rating resulted in a p-value of 0.91. In other words, there is no statistical significance in the difference in the average students' ratings of the computer lessons across the three schools.

There are a few students who said *Maybe* to wanting to do more computer science lessons in their classroom that gave a higher rating than those who said *Yes*. These results support that ratings can be subjective. This question is a nice supplement to the previous question of whether students are in favor of the computer science lessons, but asking students to rate the lessons may not be necessary.

Table 7.19: Grover Heights: Ratings of the Lessons Grouped by Answers

Rating	1	2	3	4	5	6	7	8	9	10
# of students that said "Yes"	0	0	0	0	1	2	1	2	2	9
# of students that said "Maybe"	1	0	0	0	1	0	0	0	0	3

Table 7.20: Lange: Ratings of the Lessons Grouped by Answers

Rating	1	2	3	4	5	6	7	8	9	10
# of students that said "Yes"	0	0	0	0	0	0	0	1	1	10
# of students that said "No"	0	0	0	1	2	0	0	0	0	1
# of students that said "Maybe"	0	0	0	1	3	1	0	2	1	3

Table 7.21: Pacheco: Ratings of the Lessons Grouped by Answers

Rating	1	2	3	4	5	6	7	8	9	10
# of students that said "Yes"	0	0	0	0	0	0	1	0	1	5
# of students that said "Maybe"	1	0	0	0	1	0	3	0	0	1

7.1.4 Challenges

Having limited time in the classroom was one of the main challenges. Time had to be carefully and precisely managed because each classroom had a specific time slot during which they could use the computers. Ideally, the surveys would be taken outside of the lesson time so that the whole 50-minute period could be dedicated for the lesson.

Not being able to control student attendance is another challenge of this experiment. The results may not be as precise as one had hoped because not all students were present for all four sessions. As long as they were there for the pre-survey and post-survey, they were included in the results.

Lastly, unexpected technical difficulties came up during the first classroom visit at Lange Elementary School. It was discovered that the student accounts could not access the provided Google spreadsheet necessary for the Geography lesson due to the district restrictions. A workaround was quickly devised within five minutes by creating a copy of the provided spreadsheet using the teacher's account; even so, the experience may have caused some confusion for the students.

7.1.5 Conclusion

Overall, it seems that students learned over time to undo their mistakes and try again when their changes did not work. Students generally were also able to help their classmates fix their code if they had encountered the same mistake in the past.

As the sessions went on, students worked more efficiently and showed improvements in being more detail oriented. Their improvements allowed the sessions to move along smoother and faster. This experiment took place within a relatively short period of time which probably was not enough time to reach out to all of the students. However, by the end of the experiment, one student from Grover Heights and one student from Lange each indicated that they want to become a programmer when they grow up. Although surveying young students is difficult and yields some inconclusive quantitative results, what we can still draw from the results is that students at this age range are capable of learning extremely quickly and are open to different interests.

The lessons received very positive feedback from all of the teachers. Two of the three teachers expressed excitement that the lessons in the computer science unit align extremely well with what they are teaching. The Geography and Graphs lessons were coincidentally taught during the same week that those teachers were teaching similar content in their classroom. One teacher shared that some of her students thought that these computer science lessons were their favorite compared to their previous coding activities. The principal at Lange Elementary School also provided extremely positive feedback for the computer science lessons. After his visits to the classroom to observe the Writing session and the Graphs session, he expressed appreciation for the text-based programming approach and integration of computer science into the classroom curriculum.

The lessons in the Integrated Computer Science Unit are flexible so that they can be used in the classroom and in an after school program. With that said, it may be more effective to include these activities during normal school hours because not every school has an after school program and not every student attends if there were an after school program at their school. Including these lessons in the classroom guarantees equal chances of exposure to computer science for all students.

7.2 Teaching Tool

This section discusses the survey, participants, and results of an early version of the teaching tool. The goal of this process is to gain insight into whether the provided instructions are effective in walking the teachers from start to end to set up the lesson and to understand the student activities.

7.2.1 Survey

Each participating teacher attempted to walk through the instructions to replicate a Geography lesson with their desired map and data, then filled out a survey afterwards to provide their demographic information, interest level in incorporating computer science in their classroom, and feedback on the tool. No surveying is needed for the Writing lesson because there is no set up required. The student activities are very similar to those in the Geography lab, so if teachers feel confident about the Geography lab then they should not experience problems with the Writing lab.

7.2.2 Participants

The participants that tested and gave feedback on the tool are the three teachers from Grover Heights Elementary School, Lange Elementary School, and Pacheco Elementary School, who participated in the Integrated Computer Science Unit, and two teachers from other elementary schools. Table 7.22 displays demographic information about the teachers.

Table 7.22: Participating Teachers' Demographics

Teacher #	**Grade Level**	**Subjects**	**# of Years Teaching**	**Technical Background**
1	5	Math, Language Arts	1	No experience with programming
2	4	Math, Science, History-Social Science, Language Arts, Art/Music	3-5	No experience with programming
3	5	Math, Science, History-Social Science, Language Arts, Art/Music	3-5	No experience with programming
4	4-6	Language Arts	3-5	Took one programming class in Javascript
5	6	Math, Science	3-5	Took one programming class in C++

7.2.3 Results

Teachers' Success Setting Up Lesson

Initially, teacher #1 was not able to set up the map lesson. Based on her feedback, it was most likely due to a problem with linking her Google spreadsheet into the source code. Similarly, teacher #2 also came across the same problem, which indicated that there may be a need for more clarifications in the instructions. Fortunately, direct observation was possible with this teacher, which revealed that there was an image on the website that did not render and was the cause of the confusion. After the image was resolved, teacher #2 attempted the same step again and was able to set up the lesson and completed the student activities. The total time it took her to set up, including resolving the confusion, was about 30 minutes. After circling back to teacher #1 with the updated website, she agreed to give it another try. She was also able to successfully set up the lesson and followed the student activities.

Moving forward, teachers #3 and #5 were able to set up the lesson without encountering any problem. Lastly, teacher #4 was also able to set up the lesson, although she indicated that it was confusing when she was first faced with an error message ("Error loading data from Google spreadsheet") in the Result panel. The error message is expected because the original code does not provide a Google spreadsheet yet, as shown in Listing 6.1. In order to render a map, teachers would have to paste in their Google spreadsheet key. This confusion could easily be alleviated by changing the error message to clarify that teachers will first have to provide their Google spreadsheet key before seeing a map.

Free Response Feedback

The highlights of the teachers' feedback are presented in Table 7.23. Most teachers mentioned in their feedback that the provided instructions are clear and the visuals are helpful. Teacher #1 mentioned the helpfulness of the suggestions for anticipated mistakes in the directions. Teacher #5 indicated that the instructions were easy to follow and the code was easy to replace. He perhaps found the tasks to be easy because of his prior programming experience.

Although the question asked for feedback on what they would like to be changed or added, only teacher #4 had feedback for improvement (discussed at the end of the *Teachers' Success Setting Up Lesson*).

How Teachers Feel about Integrating Computer Science

This section presents a series of three questions regarding how teachers feel about integrating computer science in their classroom. Integrating computer science (CS) in the classroom is defined on the survey as "The context of integrating computer science in your classroom refers to using coding as a tool to help your lessons (whether it is Math, Science, Language Arts, History, Music, Art, etc.) become more interactive in order to spark students' interest in computer science." Each question is scored on a scale of one to five, where one is low confidence and five is high confidence.

The first question in this series asked teachers how likely they are to integrate CS in the classroom by themselves. The results are shown in Figure 7.7, where two teachers rated that they are at a four, two rated themselves at a three, and one rated themselves at a two. The spread in the answers makes sense because teachers with different programming experience would feel differently about leading computer science integrated lessons by themselves.

Table 7.23: Highlights of Teachers' Feedback on Teaching Tool

Teacher #	Free Response Feedback
1	"The directions were very explicit. I appreciated the visuals, and the anticipated mistakes (if this doesn't work, try this.) It worked!"
2	"Since I have little to no experience with coding programs, the website did provide simple and easy to follow directions to try and set up the map. The screen shots included in the directions are very helpful."
3	"What I have done in the past is just code.org and you coming in to teach. I'd love to integrate more if I have a step by step lesson so that I know how to guide my students in learning computer science."
4	"I was able to set up the lesson, but it was confusing that the js fiddle page showed an error message in the map quadrant until I pressed Run."
5	"The instructions for setting up a Google spreadsheet were easy to follow because the visuals provided on the website matched with what I was seeing as a participant. I found it easy to replace the code in the html and javascript panels by following the instructions provided."

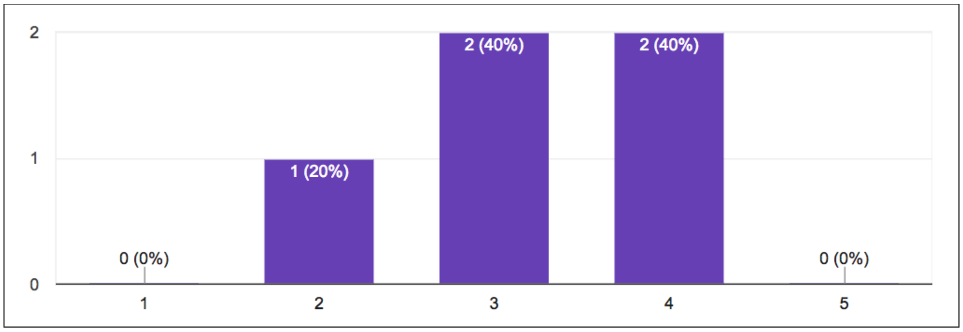


Figure 7.7: How Likely are Teachers to Integrate CS by Themselves

The second question in this series of questions asked teachers how likely they are to integrate CS in the classroom if someone were to come in to teach for them. The results are shown in Figure 7.8, where four of the five teachers answered this question with a five, meaning they are extremely likely to integrate CS if they had someone's help in leading the lessons. One teacher answered this question with a two, which makes sense if they prefer to learn the materials and lead their classroom themselves. Comparing these results with the results from Figure 7.7, it is not at all surprising that teachers feel more comfortable having someone experienced teach the lessons rather than them having to teach the lessons themselves.

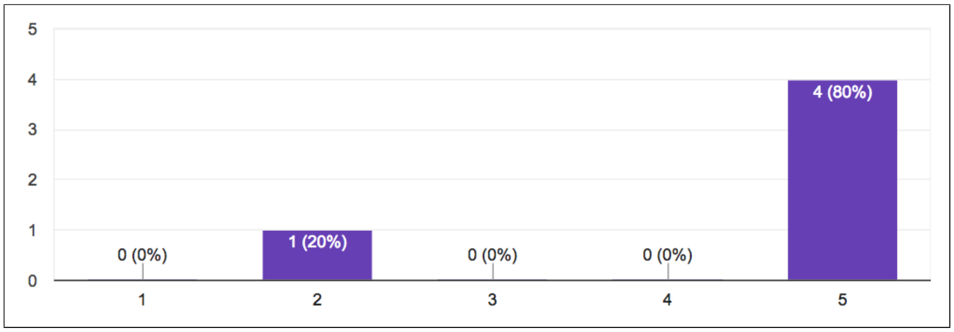


Figure 7.8: How Likely are Teachers to Integrate CS with Someone's Help

The last question in this series of questions asked teachers how likely they are to integrate CS in the classroom if they had a tool to help with lesson planning. The results are shown in Figure 7.9 where two teachers answered with a five and three teachers answered with a four. Comparing these results with the results from Figure

7.8 show that more teachers would prefer to have someone who is experienced come in and teach the computer science integrated lessons, but comparing to the results from Figure 7.7 suggest that teachers would be more likely to integrate computer science in their classroom if they had a tool to guide them.

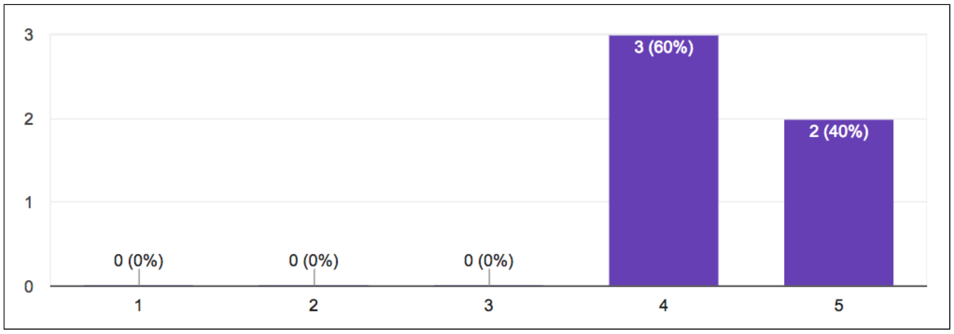


Figure 7.9: How Likely are Teachers to Integrate CS with a Tool

Table 7.24 summarizes the results grouped by teacher. Of the two teachers that indicated that they are a level four of likeliness to integrate computer science by themselves, one has one class worth of Javascript experience and one does not have any programming experience. The latter teacher's response is surprising, because it is expected that someone with no programming experience would not be as willing to teach computer science integrated lessons themselves. A speculation is that there may be some bias in her answer. The question was intended to be answered with no previous context on the Integrated Computer Science Unit. But since she had already observed the computer science lessons being taught to her students during the study, she may believe that they are straightforward enough for her to replicate on her own without any help. It is somewhat surprising that the teacher with one class worth of experience in C++ is only a level two of likeliness to integrate computer science lessons by himself. But this could be explained by him not having the context of the Integrated Computer Science Unit like some of the other teachers did and assumed that he would have to come up with these integrated computer science lessons from scratch just as the survey question intended.

Table 7.24: Teachers' Likeliness of Integrating CS given Different Methods

Teacher #	Technical Background	By Themselves	With Someone's Help	With Tool
1	No experience with programming	3	5	4
2	No experience with programming	4	5	4
3	No experience with programming	3	5	5
4	Took one programming class in Javascript	4	5	4
5	Took one programming class in C++	2	2	5

7.2.4 Challenges

The initial challenge was to determine how much detail should go into the instructions of the tool. Over-explaining could risk having long paragraphs of instructions, which defeats the purpose of the tool being easy to understand and could discourage users. Under-explaining could risk the instructions being hard to follow for someone with little to no technical background. Finding a balance was important to create a tool that is easy to use and easy to understand.

Another challenge was finding teachers who were willing to test the tool within the given timeline. The tasks and survey were estimated to take about 20-30 minutes.

The three teachers who opened up their classroom for the computer science unit were asked to test the tool during one of the four sessions. The remaining participants sent in feedback on their own time. The total time it took to receive feedback from all participants was 10 weeks.

7.2.5 Conclusion

Several of the teachers had no programming background but were still able to follow instructions from the tool to generate code for their desired lesson. All of the participating teachers indicated that they would be more willing to lead computer science lessons if given a tool to help. All of the feedback expressed appreciation for the images that accompanied the instructions and the suggested solutions for anticipated mistakes. The principal at Lange Elementary School was interested when he saw the tool being tested. He gave positive feedback regarding the benefits of the tool and asked if it would be made available to the teachers at his school. He had high hopes that teachers will give the computer science lessons a chance in their classroom.

Chapter 8

FUTURE WORK

8.1 Integrated Computer Science Unit

This section discusses future contributions that can be added to the computer science unit. The vision for moving forward includes adding more lessons, obtaining more data, and using different surveying techniques.

8.1.1 Integrating With More Subjects

Many students indicated on the surveys that their favorite subject is PE and/or Music/Art. In order to reach the interest of those students, it would be beneficial to add computer science lessons that integrate subjects of those interests as well.

8.1.2 Changing Surveying Technique

To test students' knowledge in computer science, future surveys can add a few additional questions regarding coding concepts from the lessons. To test students' interest in computer science, we may see better results if we were to ask students a series of short term questions rather than long term questions (e.g. "What job do you want to have when you grow up?" and "Are you considering working with technology in the future?").

It may be beneficial to conduct in-person interviews to follow up with the written surveys. More open-ended questions can be asked at this time since it may be easier for students to talk than to type their answers.

8.1.3 Visiting More Classrooms

Expanding the research to a larger set of students can help with data collection. More conclusive results could potentially be drawn from having a larger sample size.

8.1.4 Extending to Middle School

Although much of the discussion in this thesis work was focused on primary school, there are benefits in extending the work into the middle school level. Students can continue exploring computer science further to learn more about its applications and career choices. The lessons can include more challenging programming tasks to adjust the difficulty level for older students.

8.2 Teaching Tool

This section discusses further work that can be done to expand the tool. Future contributions include adding more integrated computer science lessons, improving guidance instructions, and improving and validating the screensharing feature.

8.2.1 Additional Lessons

As more lessons are developed and added to the Integrated Computer Science Unit, they should also get added to the tool. The outlook of this tool is to provide guidance for a collection of computer science lessons that integrate all subjects taught in school.

8.2.2 Guidance for Further Explorations

The current tool provides guidance to help teachers create the simplest lessons. More instructions can be added to guide teachers who want to explore the code further.

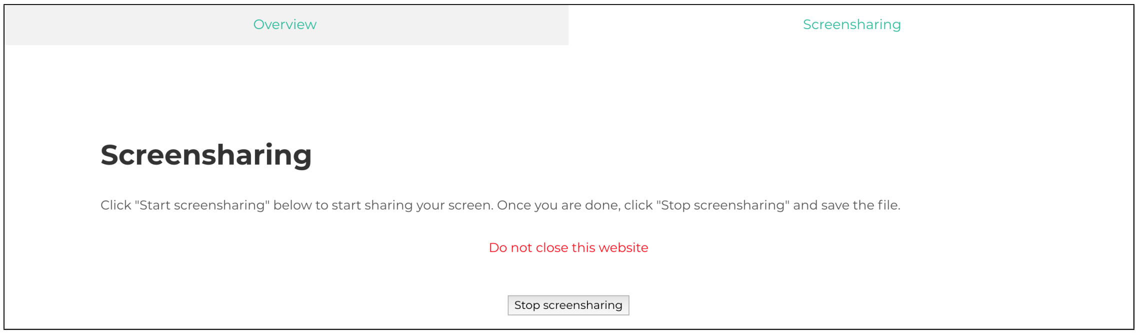
Using the Graphs lesson as an example, future instructions can show teachers how to add more slices to the existing pie graph and how to generate other types of graphs.

8.2.3 Launching Tool

There was strong indication from teachers' feedback that they would be more likely to incorporate computer science in their classroom if given a tool to help. It would be beneficial to launch the tool and observe teachers leading the lessons that they obtained from it. Results from the observations could be used to further improve the tool by adding more appropriate guidance. Lange Elementary School could be a starting point since there is interest in the tool at the school.

8.2.4 Improving Screensharing Feature

This feature currently requires students to click *Stop Recording*, shown in Figure 8.1, once they are done with the lab and click *Save* to save the screenshare recording. If a student were to mistakenly close the page or exit out of the *Save* screen prior to saving, their recording will be lost. Improvement to this feature can be made by adding guards to prevent students from losing their recording. One example would be having an alert box pop up while a recording is in session whenever the user is attempting to close the page.



Screensharing

Click "Start screensharing" below to start sharing your screen. Once you are done, click "Stop screensharing" and save the file.

Do not close this website

Figure 8.1: Stop Recording

8.2.5 Validating Screensharing Feature

No validation was done on the screensharing feature because it was developed after the lessons were over. Future classroom visits should incorporate this feature, shown in Figure 6.2, in order to obtain additional data aside from survey results.

Chapter 9

CONCLUSION

This thesis discusses two components of exposing young students to computer science. The first component is to consider how the lessons can spark student interests and develop lessons that are versatile to be integrated with different standards. The second component is to consider how to enable teachers to feel more comfortable integrating computer science in their classroom. This thesis created the foundation to help teachers begin the integration of computer science in their classroom.

The Integrated Computer Science Unit reduces the time management burden for teachers and is enjoyed by most students. The Teaching Tool helps teachers feel more confident and more willing to include computer science in their classroom. There is still much more to be done for the future of integrating computer science in K-12 schools, but the results of this thesis work provide a promising outlook.

The future of diversity in tech could improve if more teachers are willing to include computer science in their classroom and allow young students to be exposed to computer science more often and more consistently.

BIBLIOGRAPHY

[1] Adding coding to the curriculum. https://www.nytimes.com/2014/03/24/world/europe/adding-coding-to-the-curriculum.html?_r=0. Accessed: 2018-03-08.

[2] Airbnb equal employment opportunity. https://blog.atairbnb.com/wp-content/uploads/2015/10/EEO-1-2015-2.pdf?3c10be. Accessed: 2018-01-09.

[3] Airbnb diversity page. https://blog.atairbnb.com/creating-a-world-where-everyone-can-belong/. Accessed: 2018-01-09.

[4] Amazon diversity groups. https://www.amazon.com/b?node=10080092011. Accessed: 2018-01-23.

[5] Apple equal employment opportunity. https://www.scribd.com/document/333723038/Apple. Accessed: 2018-01-09.

[6] Apple diversity page. https://www.apple.com/diversity/. Accessed: 2018-01-09.

[7] Apple inclusion scholarships. https://9to5mac.com/2014/09/02/apple-offers-10k-product-integrity-inclusion-and-diversity-scholarship-for-minorities-in-tech/. Accessed: 2018-01-23.

[8] Bringing computational thinking to k-12. https://dl-acm-org.ezproxy.lib.calpoly.edu/citation.cfm?doid=1929887.1929905. Accessed: 2018-03-08.

[9] Code combat. https://codecombat.com/. Accessed: 2018-03-07.

[10] About code.org. https://code.org/about. Accessed: 2018-01-31.

[11] Code master. https://www.thinkfun.com/products/code-master/. Accessed: 2018-03-07.

[12] Code monkey planet. http://codemonkeyplanet.com/. Accessed: 2018-03-07.

[13] code.org. https://code.org/. Accessed: 2018-01-25.

[14] Code.org statistics. https://code.org/about/2015. Accessed: 2018-03-11.

[15] W3schools color picker. https://www.w3schools.com/colors/colors_picker.asp. Accessed: 2018-02-14.

[16] About common core standards. http://www.corestandards.org/about-the-standards/. Accessed: 2018-01-20.

[17] Computing degree statistics. https://nces.ed.gov/programs/digest/d16/tables/dt16_325.35.asp. Accessed: 2018-01-23.

[18] Cross-curricular definition. https://www.eduplace.com/rdg/res/literacy/interd0.htmlhttps://www.thoughtco.com/cross-curricular-connections-7791. Accessed: 2018-01-26.

[19] Cross-curricular examples. https://www.thoughtco.com/cross-curricular-connections-7791. Accessed: 2018-01-26.

[20] Computer science standards timeline. https://www.cde.ca.gov/be/st/ss/cacompsciencetimeline.asp. Accessed: 2018-05-18.

[21] Dash robot. https://www.makewonder.com/dash. Accessed: 2018-02-02.

[22] Diversity graphs article. https://gizmodo.com/the-alarming-downsides-to-tech-industry-diversity-repor-1789797486. Accessed: 2018-01-09.

[23] English standards. https://www.cde.ca.gov/be/st/ss/documents/finalelaccssstandards.pdf. Accessed: 2018-01-18.

[24] Erase all kittens. https://eraseallkittens.com/. Accessed: 2018-03-07.

[25] Facebook equal employment opportunity. https://fbnewsroomus.files.wordpress.com/2016/07/eeo-1_2015.png. Accessed: 2018-01-09.

[26] Facebook diversity page. https://newsroom.fb.com/news/2015/06/driving-diversity-at-facebook/. Accessed: 2018-01-09.

[27] What is the impact of gender diversity on technology business performance: Research summary. https://www.ncwit.org/resources/what-impact-gender-diversity-technology-business-performance-research-summary. Accessed: 2018-05-18.

[28] Gendered bias in stem toys. https://www.asee.org/public/conferences/56/papers/14121/download. Accessed: 2018-03-07.

[29] Grace hopper sponsors.

https://ghc.anitab.org/2017-sponsorships/corporate-sponsors/. Accessed: 2018-01-23.

[30] Google classroom. https://classroom.google.com/u/0/h. Accessed: 2018-01-25.

[31] Google diversity page. https://diversity.google/commitments/. Accessed: 2018-01-09.

[32] Google diversity groups. https://www.google.com/diversity/at-google.html. Accessed: 2018-01-23.

[33] Grover heights elementary school, national center for education statistics - common core of data. https://nces.ed.gov/globallocator/sch_info_popup.asp?Type=Public&ID=062308003520. Accessed: 2017-12-08.

[34] Grover heights elementary school, ed-data. http://www.ed-data.org/school/San-Luis-Obispo/Lucia-Mar-Unified/Grover-Heights-Elementary. Accessed: 2017-12-08.

[35] Diversity in high tech. https://www.eeoc.gov/eeoc/statistics/reports/hightech/. Accessed: 2018-02-17.

[36] History-social science standards. https://www.cde.ca.gov/be/st/ss/documents/histsocscistnd.pdf. Accessed: 2018-01-10.

[37] Holes by louis sachar. https://www.scholastic.com/teachers/books/holes-by-louis-sachar/. Accessed: 2018-03-07.

[38] Kahoot! https://kahoot.it/. Accessed: 2018-03-14.

[39] Lange elementary school, national center for education statistics - common core of data. https://nces.ed.gov/globallocator/sch_info_popup.asp?Type=Public&ID=062308011785. Accessed: 2017-12-08.

[40] Lange elementary school, ed-data. http://www.ed-data.org/school/San-Luis-Obispo/Lucia-Mar-Unified/Lange-(Dorothea)-Elementary. Accessed: 2017-12-08.

[41] Lego mindstorms. https://www.lego.com/en-us/mindstorms. Accessed: 2018-03-07.

[42] Standard lesson structure. http://teachingasleadership.org/sites/default/files/Related-Readings/IPD_Ch5_2011.pdf. Accessed: 2018-01-21.

[43] littlecodr. http://littlecodr.com/. Accessed: 2018-03-07.

[44] Math/history-social science integration. https://mathteachingstrategies.wordpress.com/2009/03/24/integration-of-math-and-social-studies/. Accessed: 2018-01-26.

[45] Math/science integration. http://beyondpenguins.ehe.osu.edu/issue/weather-and-climate-from-home-to-the-poles/integrating-science-and-math-data-analysis-and-weather. Accessed: 2018-01-26.

[46] Math standards. https://www.cde.ca.gov/be/st/ss/documents/ccssmathstandardaug2013.pdf. Accessed: 2018-01-11.

[47] Microsoft diversity page. https://www.microsoft.com/en-us/diversity/inside-microsoft/default.aspx#epgDivFocusArea. Accessed: 2018-01-09.

[48] Microsoft inclusion scholarships. https://careers.microsoft.com/students/scholarships. Accessed: 2018-01-23.

[49] Next generation science standards. https://www.nextgenscience.org/. Accessed: 2018-03-13.

[50] Number of computers in school. https://nces.ed.gov/pubs2010/2010034.pdf. Accessed: 2018-01-26.

[51] Pacheco elementary school, national center for education statistics - common core of data. https://nces.ed.gov/globallocator/sch_info_popup.asp?Type=Public&ID=063480005877. Accessed: 2017-12-08.

[52] Pacheco elementary school, ed-data. http://www.ed-data.org/school/San-Luis-Obispo/San-Luis-Coastal-Unified/Pacheco-Elementary. Accessed: 2017-12-08.

[53] Impact of parental attitudes. https://www.edweek.org/ew/articles/2007/10/24/09dads.h27.html. Accessed: 2018-03-07.

[54] The role of parental occupation. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3068971. Accessed: 2018-02-17.

[55] Parents and engineering knowledge.

https://www.linkedin.com/pulse/parents-need-help-educate-children-engineering-careers-mark-lynch/. Accessed: 2018-02-17.

[56] Quality of worklife survey. http://www.aft.org/sites/default/files/worklifesurveyresults2015.pdf. Accessed: 2018-03-11.

[57] California assessment of student performance and progress (caaspp). https://caaspp.cde.ca.gov/sb2016/search. Accessed: 2017-12-08.

[58] Science standards. https://www.cde.ca.gov/be/st/ss/documents/sciencestnd.pdf. Accessed: 2018-01-10.

[59] About scratch. https://scratch.mit.edu/about. Accessed: 2018-01-29.

[60] About swift playgrounds. https://www.apple.com/swift/playgrounds/. Accessed: 2018-02-01.

[61] Swift playgrounds toys. https://www.apple.com/shop/accessories/all-accessories/toys-games?page=1. Accessed: 2018-02-01.

[62] Teachers' steoretypical biases. http://www.nber.org/papers/w20909. Accessed: 2018-03-07.

[63] Tynker. https://www.tynker.com/. Accessed: 2018-03-07.

[64] Occupational outlook handbook, software developers. https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm. Accessed: 2018-01-09.

[65] Women in computer science. https://www.computerscience.org/resources/women-in-computer-science/. Accessed: 2018-01-23.

[66] Wonder workshop lessons. https://education.makewonder.com/curriculum. Accessed: 2018-02-02.

[67] Yahoo equal employment opportunity. https://s.yimg.com/ge/pr/Yahoo_EEO1-July_2015_Summary.pdf. Accessed: 2018-01-09.

[68] Yahoo diversity page. https://yahoo.tumblr.com/post/123472998984/please-see-here-for-our-eeo-1-report. Accessed: 2018-01-09.

APPENDICES

Appendix A

TEACHING TOOL DOWNLOADABLE DOCUMENTS

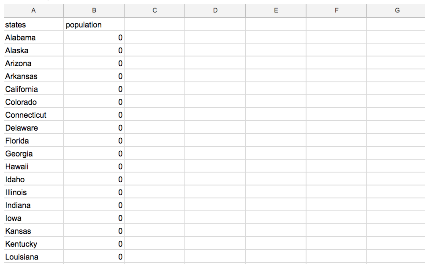
Appendix A includes the student worksheets and the teacher guides for the Geography lesson and Writing lesson.

Activity Worksheet

Name: ______________________________

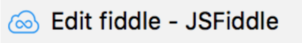
US Geography & Population

Step 1: Navigate to the Google spreadsheet and type each of your state's population in the Google spreadsheet. The spreadsheet should look something like this.

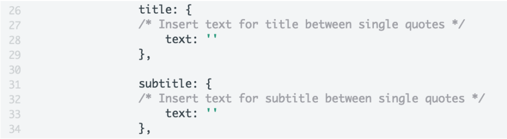


A	B	C	D	E	F	G
states	population					
Alabama	0					
Alaska	0					
Arizona	0					
Arkansas	0					
California	0					
Colorado	0					
Connecticut	0					
Delaware	0					
Florida	0					
Georgia	0					
Hawaii	0					
Idaho	0					
Illinois	0					
Indiana	0					
Iowa	0					
Kansas	0					
Kentucky	0					
Louisiana	0					

Step 2: Go to JSFiddle where your code is.

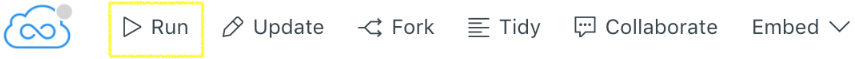


Step 4: Go to line 28 and add a title to your map. Go to line 33 and add a subtitle to your map. **Keep the single quotation marks!**

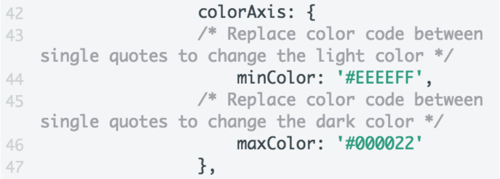


```
26                  title: {
27                  /* Insert text for title between single quotes */
28                      text: ''
29                  },
30
31                  subtitle: {
32                  /* Insert text for subtitle between single quotes */
33                      text: ''
34                  },
```

Step 5: Click 'Run' at the top left corner to see the new changes.

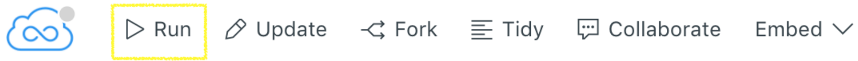


Step 6: Customize your map by changing the colors. Pick two colors from the color picker. Go to line 42 and 44, and replace the green text with your color code. **Keep the single quotation marks and the pound sign!**



```
                colorAxis: {
                /* Replace color code between single quotes to change the light color */
                    minColor: '#EEEEFF',
                /* Replace color code between single quotes to change the dark color */
                    maxColor: '#000022'
                },
```

Step 7: Click 'Run' at the top left corner to see the new changes.



Step 8: Answer these questions on a separate sheet of paper.

- Name 2 states have the highest populations. (These are states with really dark colors. You can also place your mouse over the state to see its actual population.)
- Name 2 states have the lowest populations. (These are states with really light colors. You can also place your mouse over the state to see its actual population.)

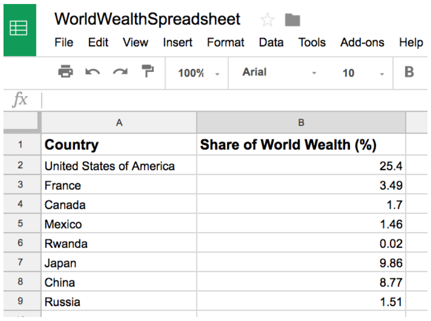
Teacher's Guide

US Geography & Population

Students will need the color chart website in order to obtain color codes to customize their map. The color chart can be found here: https://www.w3schools.com/colors/colors_picker.asp

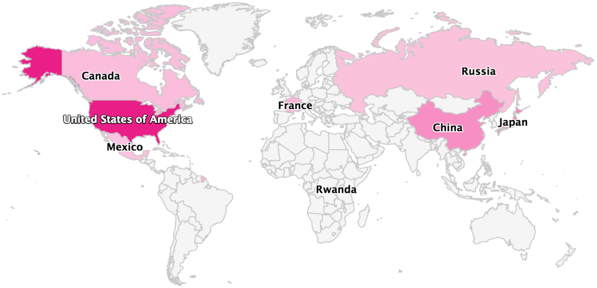
Google Spreadsheet:

- Share the Google spreadsheet with your students that you created earlier during the Set Up process.
- Instruct students to fill out the spreadsheet accordingly to your lesson.
    - For example, if your lesson is to show a map of the world and the wealth of each country, students will insert the country names in column A of the spreadsheet and the wealth number in column B of the spreadsheet.



	A	B
1	**Country**	**Share of World Wealth (%)**
2	United States of America	25.4
3	France	3.49
4	Canada	1.7
5	Mexico	1.46
6	Rwanda	0.02
7	Japan	9.86
8	China	8.77
9	Russia	1.51

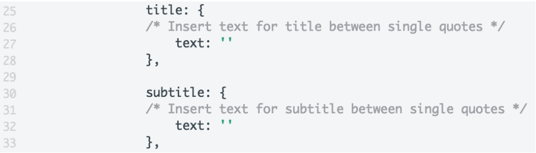
    - It may be beneficial if you fill out the locations ahead of time and only have the students fill out the corresponding numerical data since location names require correct spelling and are sensitive to case and accent marks.
    - Remember, numerical data can be whole or decimal numbers, but do not include commas.
    - Resulting map of the above spreadsheet:



Change title and subtitle:

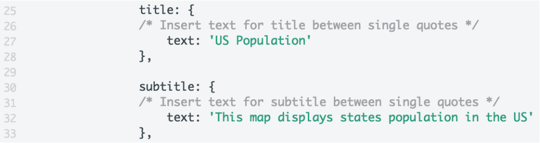
- Share the JSFiddle you set up with your students.
- Insert text in between the single quotes on line 27 and 32. You can tell this is the appropriate place to do so because line 25 indicates the text is for the title and line 30 indicates the text is for the subtitle.

Original Code:



```
25                  title: {
26                  /* Insert text for title between single quotes */
27                      text: ''
28                  },
29
30                  subtitle: {
31                  /* Insert text for subtitle between single quotes */
32                      text: ''
33                  },
```

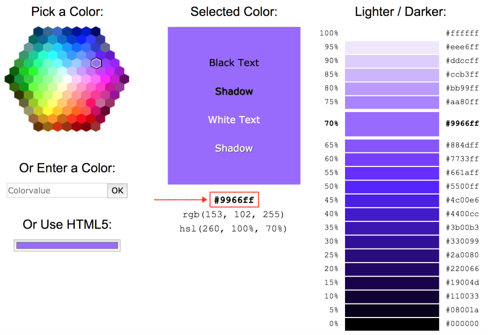
New Code:



```
25                  title: {
26                  /* Insert text for title between single quotes */
27                      text: 'US Population'
28                  },
29
30                  subtitle: {
31                  /* Insert text for subtitle between single quotes */
32                      text: 'This map displays states population in the US'
33                  },
```

Change color codes:

- Navigate to the color chart, select a color you want, highlight and copy the color code.

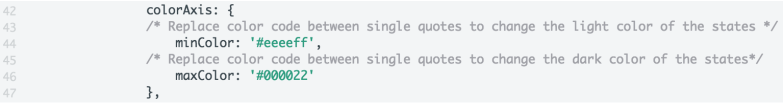


#9966ff
rgb(153, 102, 255)
hsl(260, 100%, 70%)

100%	#ffffff
95%	#eee6ff
90%	#ddccff
85%	#ccb3ff
80%	#bb99ff
75%	#aa80ff
70%	**#9966ff**
65%	#884dff
60%	#7733ff
55%	#661aff
50%	#5500ff
45%	#4c00e6
40%	#4400cc
35%	#3b00b3
30%	#330099
25%	#2a0080
20%	#220066
15%	#19004d
10%	#110033
5%	#08001a
0%	#000000

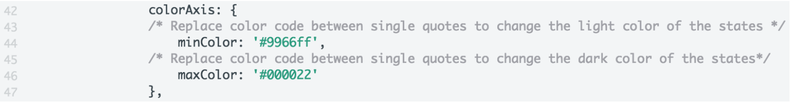
- Navigate back to JSFiddle and find line 44. Replace the color code between the single quotes by removing the old color code and pasting in the new color code.

Original Code:



```
42                  colorAxis: {
43                  /* Replace color code between single quotes to change the light color of the states */
44                      minColor: '#eeeeff',
45                  /* Replace color code between single quotes to change the dark color of the states*/
46                      maxColor: '#000022'
47                  },
```

New Code:

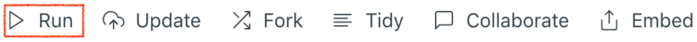


```
42                  colorAxis: {
43                  /* Replace color code between single quotes to change the light color of the states */
44                      minColor: '#9966ff',
45                  /* Replace color code between single quotes to change the dark color of the states*/
46                      maxColor: '#000022'
47                  },
```

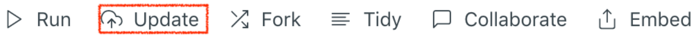
- Repeat changing colors anywhere you see color codes (pound sign followed by a 6 digit string of letters and numbers. You can find those locations on lines 22, 44, 46, 58, and 60. The gray texts surrounded by a slash / and an asterisk * as in lines 43 and 45 are called comments and will not affect the code. Reading those comments will help guide you through what it is that you are changing.

Notes:

- Depending on the students, some may find it easier to type in the color code rather than pasting in the color code.
- A good rule of thumb when changing things - do not remove surrounding syntax, such as single or double quotation marks, semicolons or colons, commas, curly brackets, etc.
- Make small changes at a time. After making a change, you should click “Run” at the top of the JSFiddle page to verify that your change is working appropriately.



- You can make changes to the code in JSFiddle and add comments to better if you want to clarify things better for your students.
  - Only texts in the comments are gray, code should not be gray. If adding comments turn some code gray, make sure you have formatted your comment correctly.
  - Once you are done and ready to save, click “Update” at the top of the JSFiddle page. It will generate a new URL with your new code. You will have to save the new URL and use that URL for future references.

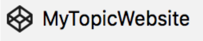


Student Worksheet

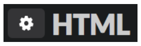
Name: ______________________________

Journalism & Making a Website

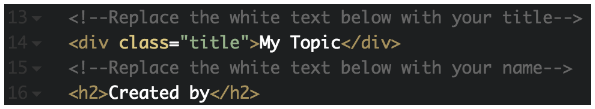
Step 1: Find the tab that says MyTopicWebsite.



Step 2: Find the HTML panel on the right side of the screen.



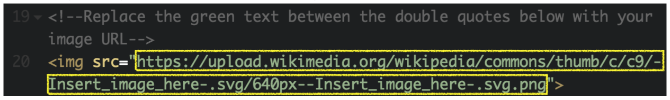
Step 3: Scroll down on the HTML panel and fill in your topic title on line 14 and your name on line 16 where the white texts are.



```
13    <!--Replace the white text below with your title-->
14    <div class="title">My Topic</div>
15    <!--Replace the white text below with your name-->
16    <h2>Created by</h2>
```

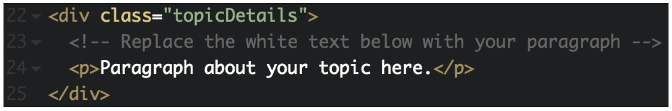
Step 4: Open a new tab and search on Google for a picture that represents your topic and copy its URL. Ask for help if you are unsure how to do this step!

Step 5: Scroll down on the HTML panel and replace the green text on line 20 with the URL by deleting the green text and pasting in the URL. **Keep the quotation marks**!



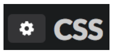
```
19  <!--Replace the green text between the double quotes below with your
    image URL-->
20  <img src="https://upload.wikimedia.org/wikipedia/commons/thumb/c/c9/-
    Insert_image_here-.svg/640px--Insert_image_here-.svg.png">
```

Step 6: Scroll down on the HTML panel and find line 24 and replace the white text with a paragraph to summarize your topic.

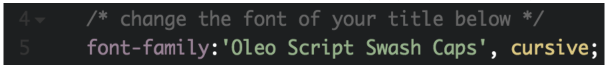


```
22  <div class="topicDetails">
23    <!-- Replace the white text below with your paragraph -->
24    <p>Paragraph about your topic here.</p>
25  </div>
```

Step 7: Find the CSS panel on the right side of the screen.

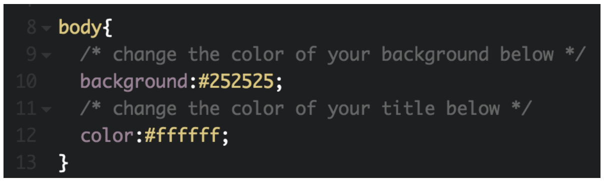


Step 8: Customize your website by changing CSS code. Refer to the fonts below to choose the font you want and copy the CSS code below the font sample. For example, you can change the font of your title by deleting the entire line 5 and pasting in the code for your new font. **Scroll down on the CSS panel and look for gray texts to tell you what the code changes!**

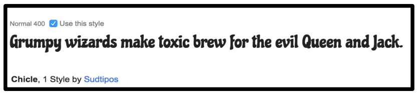


```
4    /* change the font of your title below */
5    font-family:'Oleo Script Swash Caps', cursive;
```

Step 9: Refer to the color chart website to change your colors. Replace the hex color codes with your new color code. For example, you can change the color of your background by deleting the text *#25252* on line 10 then pasting in your new color code. **Scroll down on the CSS panel and look for gray texts to tell you what the code changes!**



```
8   body{
9     /* change the color of your background below */
10    background:#252525;
11    /* change the color of your title below */
12    color:#ffffff;
13  }
```



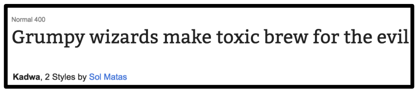
Normal 400 ☑ Use this style

Grumpy wizards make toxic brew for the evil Queen and Jack.

Chicle, 1 Style by Sudtipos

CSS:

font-family: 'Chicle', cursive;



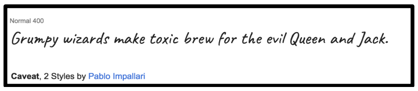
Normal 400

Grumpy wizards make toxic brew for the evil

Kadwa, 2 Styles by Sol Matas

CSS:

font-family: 'Kadwa', serif;



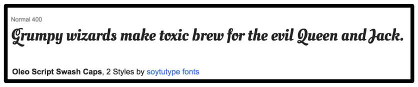
Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Caveat, 2 Styles by Pablo Impallari

CSS:

font-family: 'Caveat', cursive;



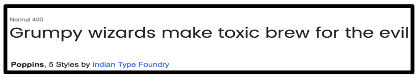
Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Oleo Script Swash Caps, 2 Styles by soytutype fonts

CSS:

font-family: 'Oleo Script Swash Caps', cursive;



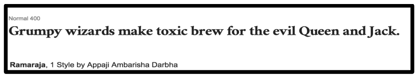
Normal 400

Grumpy wizards make toxic brew for the evil

Poppins, 5 Styles by Indian Type Foundry

CSS:

font-family: 'Poppins', sans-serif;



Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Ramaraja, 1 Style by Appaji Ambarisha Darbha

CSS:

font-family: 'Ramaraja', serif;

Teacher’s Guide

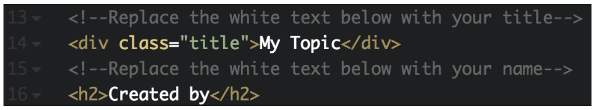
Journalism & Making a Website

Students will need the color chart website in order to obtain color codes to customize their website. The color chart can be found here: https://www.w3schools.com/colors/colors_picker.asp

CodePen:
Share the CodePen website with your students.

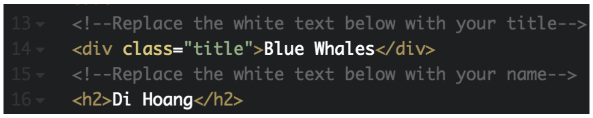
Change title and author name (HTML panel):

- Insert text on line 14 to change the topic name and line 16 to change the author name.
  Original Code:



```
13  <!--Replace the white text below with your title-->
14  <div class="title">My Topic</div>
15  <!--Replace the white text below with your name-->
16  <h2>Created by</h2>
```

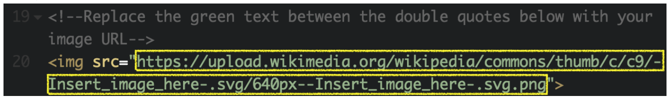
New Code:



```
13  <!--Replace the white text below with your title-->
14  <div class="title">Blue Whales</div>
15  <!--Replace the white text below with your name-->
16  <h2>Di Hoang</h2>
```

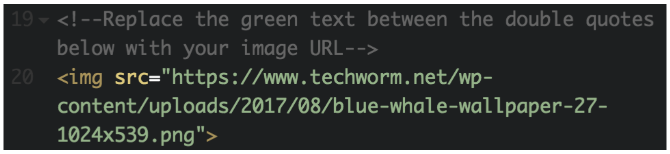
Change image URL (HTML panel):

- In order to obtain a new image, find an image on Google, right click the image, and select “Copy image address”.
- Navigate to line 20. Delete the green text in between the double quotation marks, then paste in the new image address that you just copied.
  Original Code:



```
19  <!--Replace the green text between the double quotes below with your
    image URL-->
20  <img src="https://upload.wikimedia.org/wikipedia/commons/thumb/c/c9/-
    Insert_image_here-.svg/640px--Insert_image_here-.svg.png">
```

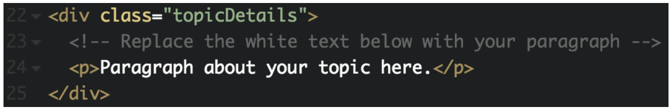
New Code:



```
19  <!--Replace the green text between the double quotes
    below with your image URL-->
20  <img src="https://www.techworm.net/wp-
    content/uploads/2017/08/blue-whale-wallpaper-27-
    1024x539.png">
```

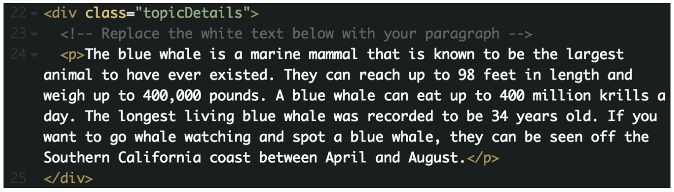
Change paragraph content (HTML panel):

- To add paragraph content into the website, navigate to line 24 and replace the white text with what you want to write.
  Original Code:



```
22  <div class="topicDetails">
23    <!-- Replace the white text below with your paragraph -->
24    <p>Paragraph about your topic here.</p>
25  </div>
```

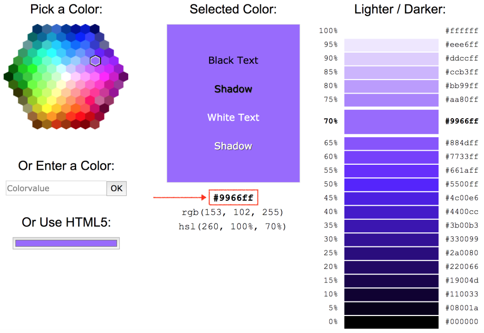
New Code:



```
22  <div class="topicDetails">
23    <!-- Replace the white text below with your paragraph -->
24    <p>The blue whale is a marine mammal that is known to be the largest
    animal to have ever existed. They can reach up to 98 feet in length and
    weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a
    day. The longest living blue whale was recorded to be 34 years old. If you
    want to go whale watching and spot a blue whale, they can be seen off the
    Southern California coast between April and August.</p>
25  </div>
```

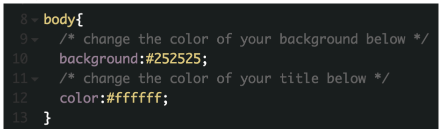
Change color codes (CSS panel):

- Navigate to the color chart, select a color you want, highlight and copy the color code.



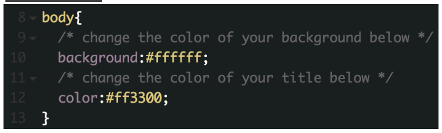
100%	#ffffff
95%	#eee6ff
90%	#ddccff
85%	#ccb3ff
80%	#bb99ff
75%	#aa80ff
70%	**#9966ff**
65%	#884dff
60%	#7733ff
55%	#661aff
50%	#5500ff
45%	#4c00e6
40%	#4400cc
35%	#3b00b3
30%	#330099
25%	#2a0080
20%	#220066
15%	#19004d
10%	#110033
5%	#08001a
0%	#000000

- Navigate back to CodePen and find line 10. Replace the color code removing the old color code and pasting in the new color code.
  Original Code:



```
8   body{
9     /* change the color of your background below */
10    background:#252525;
11    /* change the color of your title below */
12    color:#ffffff;
13  }
```

New Code:

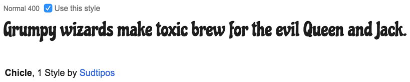


```
8   body{
9     /* change the color of your background below */
10    background:#ffffff;
11    /* change the color of your title below */
12    color:#ff3300;
13  }
```

- Repeat changing colors anywhere you see color codes (pound sign followed by a 6 digit string of letters and numbers. You can find those locations on lines 10, 12, 19, and 28. Reading gray comments will help guide you through what it is that you are changing.

Change font styles (CSS panel):

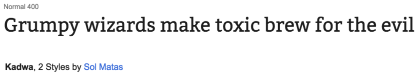
Font styles and their code:


Normal 400 ☑ Use this style

Grumpy wizards make toxic brew for the evil Queen and Jack.

Chicle, 1 Style by Sudtipos

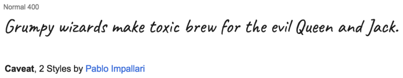
font-family: 'Chicle', cursive;


Normal 400

Grumpy wizards make toxic brew for the evil

Kadwa, 2 Styles by Sol Matas

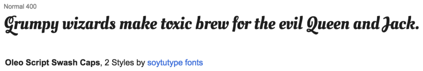
font-family: 'Kadwa', serif;


Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Caveat, 2 Styles by Pablo Impallari

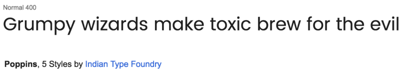
font-family: 'Caveat', cursive;


Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Oleo Script Swash Caps, 2 Styles by soytutype fonts

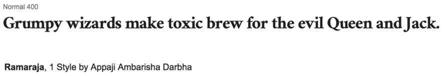
font-family: 'Oleo Script Swash Caps', cursive;


Normal 400

Grumpy wizards make toxic brew for the evil

Poppins, 5 Styles by Indian Type Foundry

font-family: 'Poppins', sans-serif;

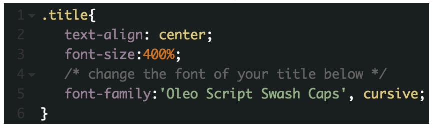

Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

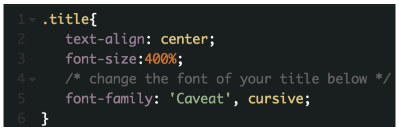
Ramaraja, 1 Style by Appaji Ambarisha Darbha

font-family: 'Ramaraja', serif;

- Changing font styles can be accomplished with CSS code. For example, you can change the font style of the title by going to line 5 and replace the entire line with the font you just copied. The left shows the original code with the 4th font style of Oleo Script Swash Caps, the right shows the new code with the 3rd font style of Caveat.



```
1 .title{
2     text-align: center;
3     font-size:400%;
4     /* change the font of your title below */
5     font-family:'Oleo Script Swash Caps', cursive;
6 }
```



```
1 .title{
2     text-align: center;
3     font-size:400%;
4     /* change the font of your title below */
5     font-family: 'Caveat', cursive;
6 }
```

- You can repeat this step on line 26 as well to change the font of the paragraph.

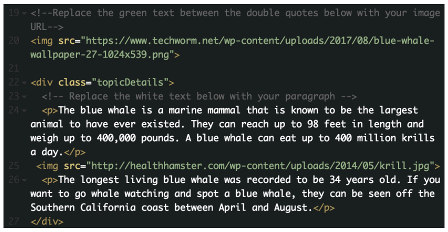
Notes:

- Depending on the students, some may find it easier to type in the color code rather than pasting in the color code.
- A good rule of thumb when changing things - do not remove surrounding syntax, such as single or double quotation marks, semicolons or colons, commas, curly brackets, etc.
- Make small changes at a time. After making a change, the page should reload automatically and you should see new content appear.

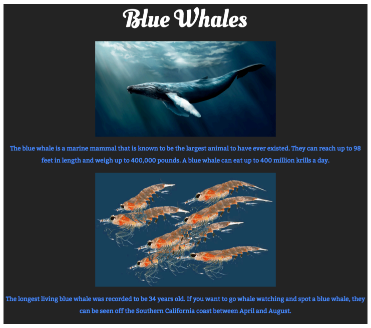
- If students want to save their website, they can click “Fork” at the top of the CodePen page. It will generate a new URL with your new code. They will have to save the new URL and use that URL for future references.

Additional/Optional Things Students Can Change:

- If students want to add more images, you can add more image tags into HTML code. Image tags look like this: <img src=“Image URL”>. For example, the below code will produce two images, one for the blue whale, one for krills. The image tag on line 20 is for the blue whale image, the image tag on line 25 is for the krills.



```
19 <!--Replace the green text between the double quotes below with your image
   URL-->
20 <img src="https://www.techworm.net/wp-content/uploads/2017/08/blue-whale-
   wallpaper-27-1024x539.png">
21
22 <div class="topicDetails">
23   <!-- Replace the white text below with your paragraph -->
24   <p>The blue whale is a marine mammal that is known to be the largest
   animal to have ever existed. They can reach up to 98 feet in length and
   weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills
   a day.</p>
25  <img src="http://healthhamster.com/wp-content/uploads/2014/05/krill.jpg">
26   <p>The longest living blue whale was recorded to be 34 years old. If you
   want to go whale watching and spot a blue whale, they can be seen off the
   Southern California coast between April and August.</p>
27 </div>
```

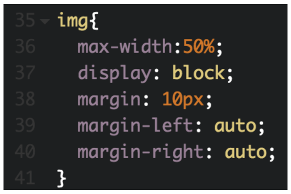


Blue Whales

The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day.

The longest living blue whale was recorded to be 34 years old. If you want to go whale watching and spot a blue whale, they can be seen off the Southern California coast between April and August.

- If students want to make their images larger or smaller, this can be done in the CSS code. You can go to line 36 in the CSS panel and change the max-width percentage number. To make an image bigger, adjust the number to be bigger. To make an image smaller, adjust the number to be smaller.



```
35 img{
36   max-width:50%;
37   display: block;
38   margin: 10px;
39   margin-left: auto;
40   margin-right: auto;
41 }
```

- If students want to make their font sizes larger, this can be accomplished in the CSS panel by changing the percentage numbers. Line 3 will change the title text size, line 17 will change the author name text size, and line 24 will change the paragraph text size.

Appendix B

IRB PROPOSAL

Appendix B includes materials that were sent to the Cal Poly IRB for approval. The following pages contain a copy of the IRB proposal, two permission forms (English and Spanish), two child assent scripts (one for the pre-survey and one for the post-survey), the pre-survey, and the post-survey.

Title
Integrated Computer Science Unit for Early Exposure to Diverse Population

Researcher and Advisor
Researcher: Di Hoang, Computer Science Department, Cal Poly San Luis Obispo
Advisor: Aaron Keen, Computer Science Department, Cal Poly San Luis Obispo

Summary
The purpose of this study is to expose young elementary school students to programming in parallel with the subjects they learn in their classroom. The goal is to spark their interest in Computer Science and increase diversity in the field of computing. I will be teaching one lesson per week and have the students apply what they've learned through basic coding. The setting of this study will take place in elementary school classrooms. There are no known risks in this study, but benefits include the students being able to learn new programming concepts that are not taught in school and the results could potentially help shape after school programs and classroom lessons. Data will be collected through two surveys.

Methods
Subjects and Subject Characteristics
The subjects of this study will be 4th-6th graders from local elementary schools. Each classroom is expected to have 20-30 students.

Investigators
Di Hoang. Experienced with volunteering for after school programs and tutoring Computer Science through a teaching assistant position.

Materials and Procedures
I will be in contact with various local elementary schools to see if I could run my study in their classroom. I will have the school principal approve the parental permission form and I will have the students take the form home for their parents to sign. Data will be collected through two surveys, a pre-survey and a post-survey.

- A pre-survey will be given on the first day of the program to gauge on student's knowledge and interest in Computer Science:
  https://www.surveymonkey.com/r/CTHCL5H
- A post-survey will be given on the last day of the program to see if their knowledge and interest have changed: https://www.surveymonkey.com/r/CJGZ5Z7

The survey is stored under my account and I will be the only one to have access to the individual results. The final study results will not include any information about each individual alone. It will be an overall trend of how each group of students favor my activities and how effective my lessons are at exposing the students as a whole to basic Computer Science concepts. No identifiable information about each group of students or each individual student will be included in the study results.

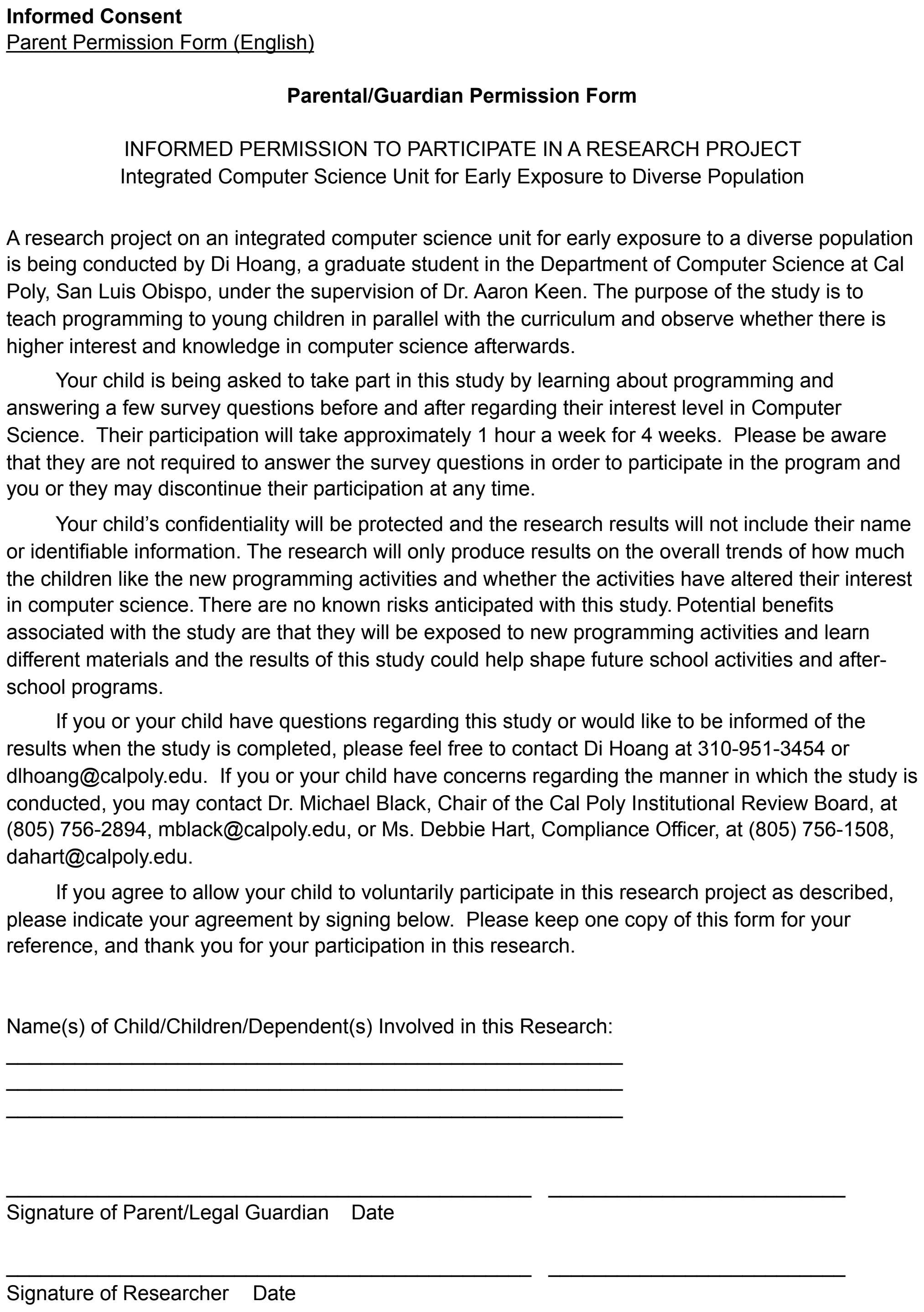
The outline of the lesson plans are included for reference. I will spend about 20 minutes teaching and giving a demo of the activity. The students will then have about 30 minutes to apply what they've learned and follow some instructions to accomplish a coding assignment.
https://docs.google.com/document/d/1eOOVWHpgjtUWNyGhUj9-xT21bLhL450oY081XQJgfpM/edit?usp=sharing

Regarding subject safety, there are no known risks in this study. The students will get dropped off at the computer lab by their teacher and get picked up at the computer lab when the session ends. Throughout the after school session, the students will only be sitting at their desk and interacting with their computer.

Study Location

One school that has agreed to this study is Pacheco Elementary School. Every quarter, Pacheco hosts an after school program called Code Club, and they are willing to execute this study as part of their after school program. The students are already signed up for the after school program for this school year. The permission form will be given to the parents of those students that are registered for the program.

There are also some schools in the Five Cities area that may be interested in running this study in their after school program, but an agreement with them has not been secured. If we come to an agreement, an update will be sent for review if necessary.



Informed Consent
Parent Permission Form (English)

Parental/Guardian Permission Form

INFORMED PERMISSION TO PARTICIPATE IN A RESEARCH PROJECT
Integrated Computer Science Unit for Early Exposure to Diverse Population

A research project on an integrated computer science unit for early exposure to a diverse population is being conducted by Di Hoang, a graduate student in the Department of Computer Science at Cal Poly, San Luis Obispo, under the supervision of Dr. Aaron Keen. The purpose of the study is to teach programming to young children in parallel with the curriculum and observe whether there is higher interest and knowledge in computer science afterwards.

Your child is being asked to take part in this study by learning about programming and answering a few survey questions before and after regarding their interest level in Computer Science. Their participation will take approximately 1 hour a week for 4 weeks. Please be aware that they are not required to answer the survey questions in order to participate in the program and you or they may discontinue their participation at any time.

Your child's confidentiality will be protected and the research results will not include their name or identifiable information. The research will only produce results on the overall trends of how much the children like the new programming activities and whether the activities have altered their interest in computer science. There are no known risks anticipated with this study. Potential benefits associated with the study are that they will be exposed to new programming activities and learn different materials and the results of this study could help shape future school activities and after-school programs.

If you or your child have questions regarding this study or would like to be informed of the results when the study is completed, please feel free to contact Di Hoang at 310-951-3454 or dlhoang@calpoly.edu. If you or your child have concerns regarding the manner in which the study is conducted, you may contact Dr. Michael Black, Chair of the Cal Poly Institutional Review Board, at (805) 756-2894, mblack@calpoly.edu, or Ms. Debbie Hart, Compliance Officer, at (805) 756-1508, dahart@calpoly.edu.

If you agree to allow your child to voluntarily participate in this research project as described, please indicate your agreement by signing below. Please keep one copy of this form for your reference, and thank you for your participation in this research.

Name(s) of Child/Children/Dependent(s) Involved in this Research:
____
____
____

______________________________________ ______________________
Signature of Parent/Legal Guardian Date

______________________________________ ______________________
Signature of Researcher Date

Parent Permission Form (Spanish)

Formulario de permiso parental / tutor

PERMISO PARA PARTICIPAR EN UN PROYECTO DE INVESTIGACIÓN
Unidad integrada de Ciencias de la Computación para la exposición temprana a una población diversa

Un proyecto de investigación sobre una unidad de ciencias de la computación integrada para la exposición temprana a una población diversa está siendo dirigido por Di Hoang, una estudiante de postgrado en el Departamento de Ciencias de la Computación en Cal Poly, San Luis Obispo, bajo la supervisión del Dr. Aaron Keen. El propósito del estudio es enseñar programación a niños de educación primaria en paralelo con el plan de estudios y observar si después hay mayor interés y conocimiento en el area de ciencias de la computación.

Se le pide a su hijo(a) que participe en este studio al aprender sobre programación, y que responda una encuesta antes y después de las clases de programación sobre su nivel de interés en ciencias de la computación. Su participación tomará aproximadamente 1 hora a la semana durante 4 semanas. Tenga en cuenta que no están obligados a responder las preguntas de la encuesta para poder participar en el programa y usted o ellos pueden descontinuar su participación en cualquier momento.

La confidencialidad de su hijo(a) estará protegida y los resultados de la investigación no incluirán el nombre o información identificable de su hijo(a). La investigación solo producirá resultados sobre las tendencias generales de cuánto le gustan a los niños las nuevas actividades de programación y si las actividades han alterado su interés en ciencias de la computación. No se conocen riesgos anticipados con este estudio. Los posibles beneficios asociados con el estudio son que los niños estarán expuestos a nuevas actividades de programación y aprenderán diferentes materiales, y los resultados de este estudio podrían ayudar a preparer/planear nuevas actividades escolares y los programas/ clases después de la escuela.

Si usted o su hijo(a) tienen preguntas sobre este estudio o si desean recibir información sobre los resultados cuando se complete el estudio, comuníquese con Di Hoang al 310-951-3454 o dlhoang@calpoly.edu. Si usted o su hijo(a) tienen dudas con respecto a la forma en que se realiza el estudio, puede comunicarse con el Dr. Michael Black, presidente de la Junta de Revisión Institucional de Cal Poly al (805) 756-2894, mblack@calpoly.edu, o la Sra. Debbie Hart, Oficial de Cumplimiento, al (805) 756-1508, dahart@calpoly.edu.

Si acepta permitir que su hijo(a) participe voluntariamente en este proyecto de investigación tal como se describe, indique su acuerdo firmando a continuación. Guarde una copia de este formulario para su referencia, y gracias por su participación en esta investigación.

Nombre (s) del niño(a) / niños(as) / dependientes involucrados en esta investigación:
____
____
____

____ _________________________
Firma del padre/tutor legal Fecha

____ _________________________
Firma del investigador Fecha

Child Assent Script (Pre-Survey)

We have come up with some new coding activities that we would like to present to you. We want to see if students like you will enjoy and learn from these coding activities, so we need your help.

We have a survey with some questions we would like to ask about you and what you know about coding. Once you are done with the survey, we will show you some awesome things you can do with coding and you can even try it out for yourself. Would you be willing to answer the survey questions?

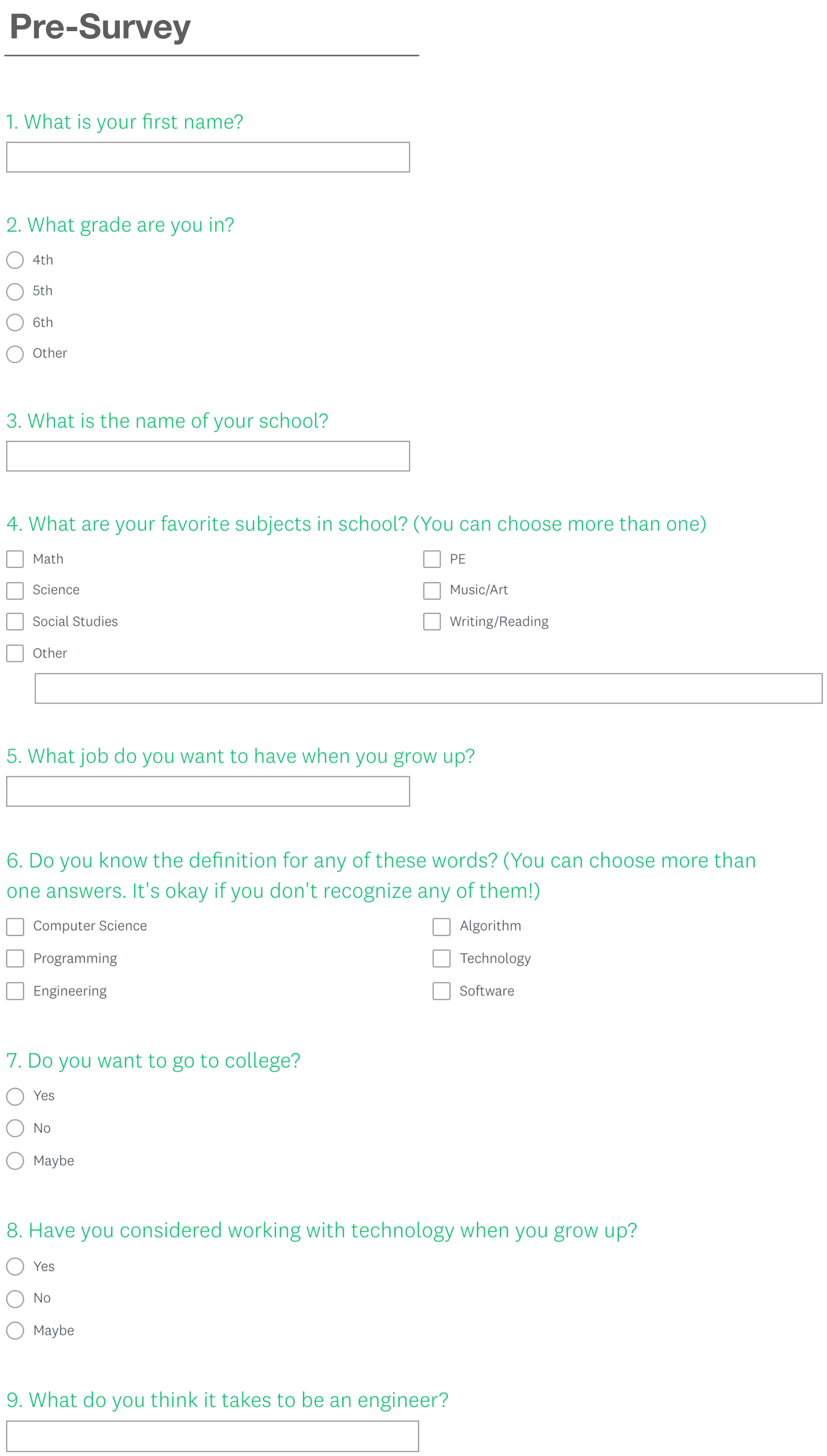
It should take only about 5 minutes and you can answer the questions as best as you can. The survey will not affect your grades in school whatsoever and if you decide at any point you do not want to continue, just let us know you want to stop.

Child Assent Script (Post-Survey)

Now that we have completed our coding program, we would like to ask you some survey questions about what you think about the activities. It should take only 10 minutes and you can answer the questions as best as you can. Would you like to help us by answering the survey questions?

Your grades in school will not be affected whether you decide to do the survey or what you answers on the survey are. If at any point you decide you don't want to continue, let us know you want to stop.

Debriefing Statement for Projects Involving Deception and Incomplete Disclosure
Not Applicable



Pre-Survey

1. What is your first name?

2. What grade are you in?

○ 4th
○ 5th
○ 6th
○ Other

3. What is the name of your school?

4. What are your favorite subjects in school? (You can choose more than one)

□ Math
□ Science
□ Social Studies
□ Other

□ PE
□ Music/Art
□ Writing/Reading

5. What job do you want to have when you grow up?

6. Do you know the definition for any of these words? (You can choose more than one answers. It's okay if you don't recognize any of them!)

□ Computer Science
□ Programming
□ Engineering

□ Algorithm
□ Technology
□ Software

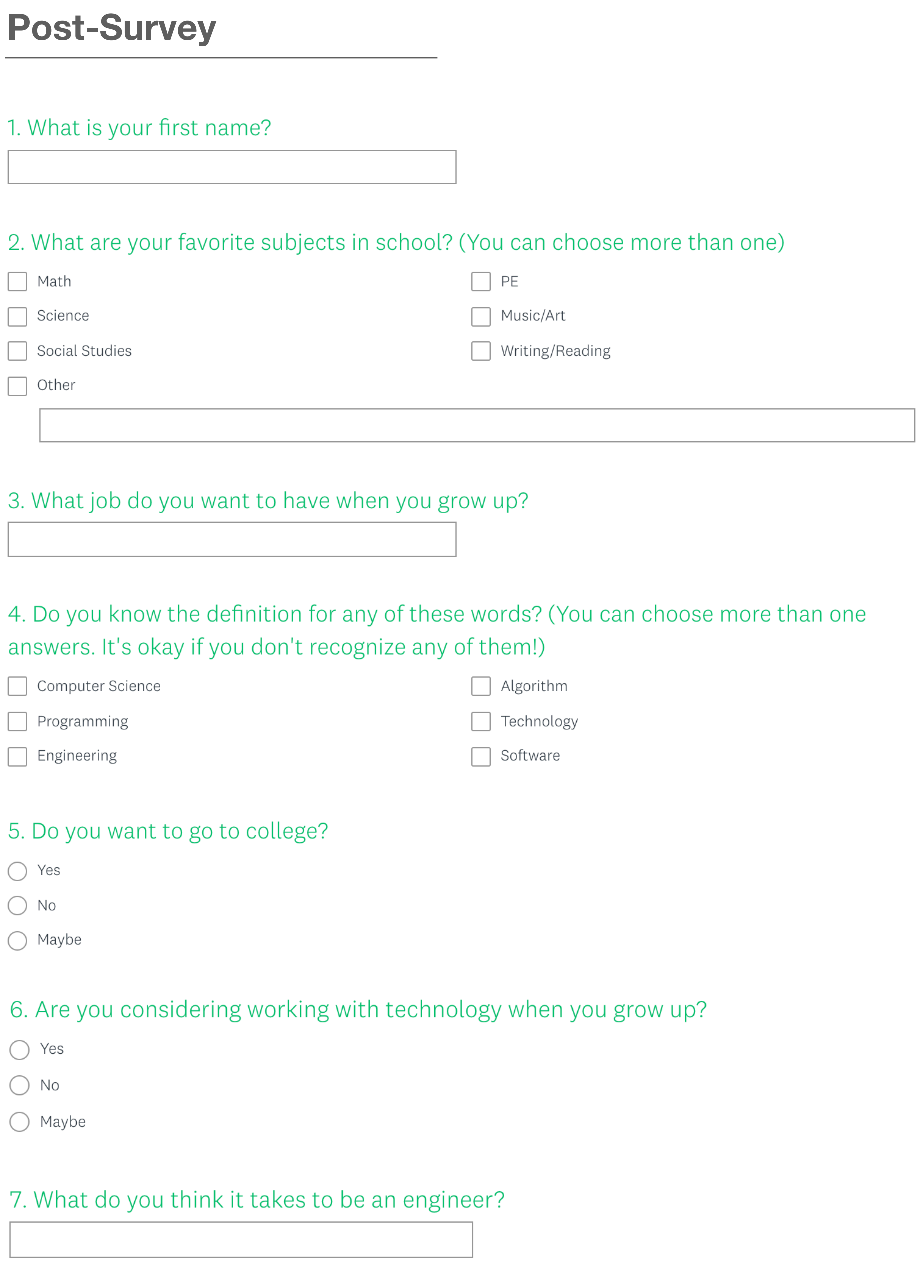
7. Do you want to go to college?

○ Yes
○ No
○ Maybe

8. Have you considered working with technology when you grow up?

○ Yes
○ No
○ Maybe

9. What do you think it takes to be an engineer?



Post-Survey

1. What is your first name?

2. What are your favorite subjects in school? (You can choose more than one)

□ Math

□ Science

□ Social Studies

□ Other

□ PE

□ Music/Art

□ Writing/Reading

3. What job do you want to have when you grow up?

4. Do you know the definition for any of these words? (You can choose more than one answers. It's okay if you don't recognize any of them!)

□ Computer Science

□ Programming

□ Engineering

□ Algorithm

□ Technology

□ Software

5. Do you want to go to college?

○ Yes

○ No

○ Maybe

6. Are you considering working with technology when you grow up?

○ Yes

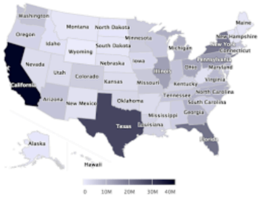
○ No

○ Maybe

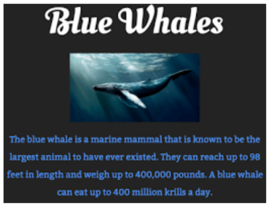
7. What do you think it takes to be an engineer?

8. Which activity was your MOST favorite and why?

1. US Geography and Population



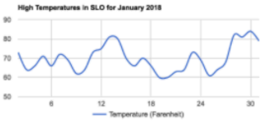
2. Journalism and Making a Website

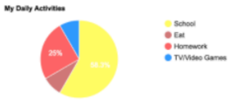


Blue Whales

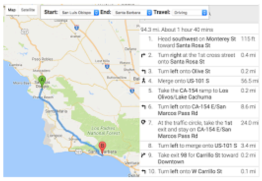
The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day.

3. Weather and Graphs





4. Geography and Google Maps API



Start: San Luis Obispo End: Santa Barbara Travel: Driving

94.3 mi. About 1 hour 40 mins

1. Head **southwest** on **Monterey St** toward **Santa Rosa St** 115 ft

2. Turn **right** at the 1st cross street onto **Santa Rosa St** 0.4 mi

3. Turn **left** onto **Olive St** 0.2 mi

4. Merge onto **US-101 S** 56.5 mi

5. Take the **CA-154** ramp to **Los Olivos/Lake Cachuma** 0.2 mi

6. Turn **left** onto **CA-154 E/San Marcos Pass Rd** 8.6 mi

7. At the traffic circle, take the **1st** exit and stay on **CA-154 E/San Marcos Pass Rd** 24.0 mi

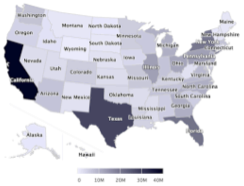
8. Turn **left** to merge onto **US-101 S** 3.4 mi

9. Take exit **98** for **Carrillo St** toward **Downtown** 0.2 mi

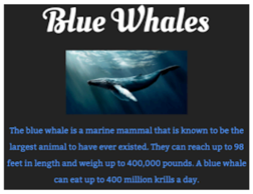
10. Turn **left** onto **W Carrillo St** 0.1 mi

9. Which activity was your LEAST favorite and why?

1. US Geography and Population



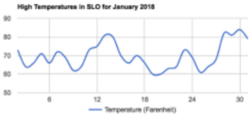
2. Journalism and Making a Website

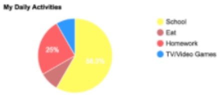


Blue Whales

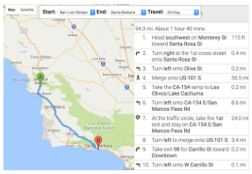
The blue whale is a marine mammal that is known to be the largest animal to have ever existed. They can reach up to 98 feet in length and weigh up to 400,000 pounds. A blue whale can eat up to 400 million krills a day.

3. Weather and Graphs





4. Geography and Google Maps API



Start: San Luis Obispo End: Santa Barbara Travel: Driving

94.3 mi. About 1 hour 40 mins

1.	Head southwest on Monterey St toward Santa Rosa St	115 ft
2.	Turn right at the 1st cross street onto Santa Rosa St	0.4 mi
3.	Turn left onto Olive St	0.2 mi
4.	Merge onto US-101 S	56.5 mi
5.	Take the CA-154 ramp to Los Olivos/Lake Cachuma	0.2 mi
6.	Turn left onto CA-154 E/San Marcos Pass Rd	8.6 mi
7.	At the traffic circle, take the 1st exit and stay on CA-154 E/San Marcos Pass Rd	24.0 mi
8.	Turn left to merge onto US-101 S	3.4 mi
9.	Take exit 98 for Carrillo St toward Downtown	0.2 mi
10.	Turn left onto W Carrillo St	0.1 mi

10. Would you like to do these activities in your classroom?

○ Yes

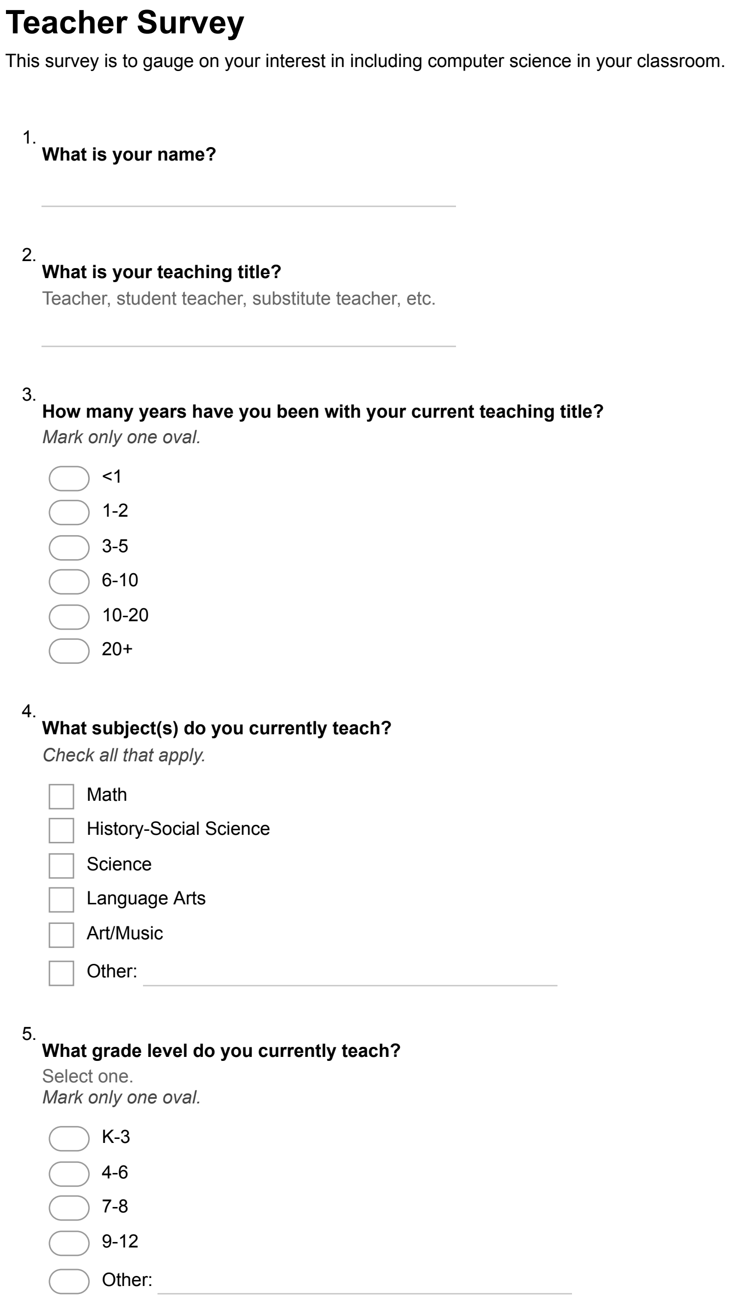
○ No

○ Maybe

11. Rate how much you've enjoyed these activities.

Not so much. Loved it!

☆ ☆ ☆ ☆ ☆ ☆ ☆ ☆ ☆ ☆



Teacher Survey

This survey is to gauge on your interest in including computer science in your classroom.

1. **What is your name?**

   ____

2. **What is your teaching title?**

   Teacher, student teacher, substitute teacher, etc.

   ____

3. **How many years have you been with your current teaching title?**

   Mark only one oval.

   ○ <1
   ○ 1-2
   ○ 3-5
   ○ 6-10
   ○ 10-20
   ○ 20+

4. **What subject(s) do you currently teach?**

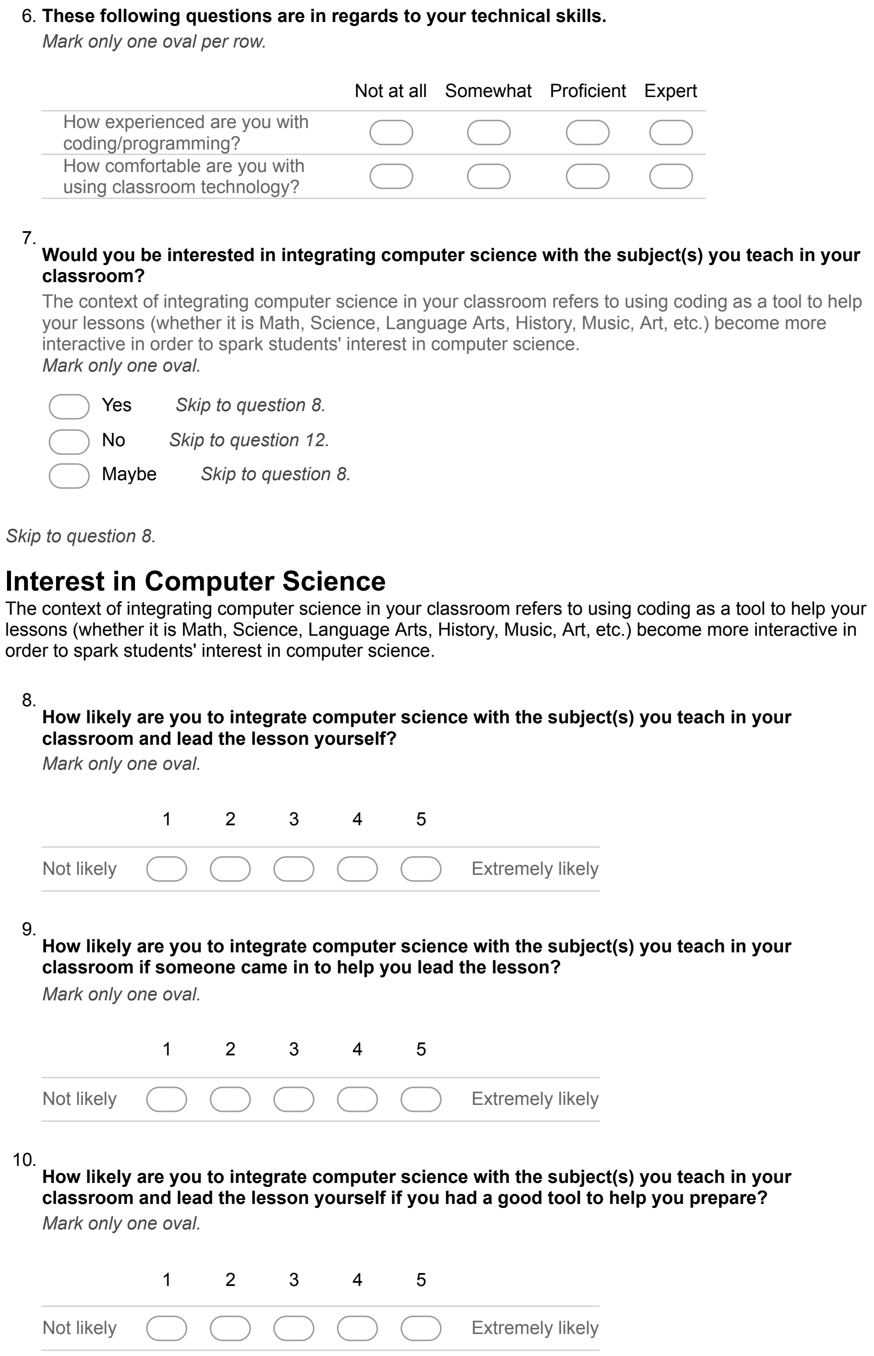
   Check all that apply.

   □ Math
   □ History-Social Science
   □ Science
   □ Language Arts
   □ Art/Music
   □ Other: ____________________

5. **What grade level do you currently teach?**

   Select one.
   Mark only one oval.

   ○ K-3
   ○ 4-6
   ○ 7-8
   ○ 9-12
   ○ Other: ____________________



6. **These following questions are in regards to your technical skills.**
Mark only one oval per row.

	Not at all	Somewhat	Proficient	Expert
How experienced are you with coding/programming?	○	○	○	○
How comfortable are you with using classroom technology?	○	○	○	○

7. **Would you be interested in integrating computer science with the subject(s) you teach in your classroom?**
The context of integrating computer science in your classroom refers to using coding as a tool to help your lessons (whether it is Math, Science, Language Arts, History, Music, Art, etc.) become more interactive in order to spark students' interest in computer science.
Mark only one oval.

○ Yes *Skip to question 8.*
○ No *Skip to question 12.*
○ Maybe *Skip to question 8.*

Skip to question 8.

Interest in Computer Science

The context of integrating computer science in your classroom refers to using coding as a tool to help your lessons (whether it is Math, Science, Language Arts, History, Music, Art, etc.) become more interactive in order to spark students' interest in computer science.

8. **How likely are you to integrate computer science with the subject(s) you teach in your classroom and lead the lesson yourself?**
Mark only one oval.

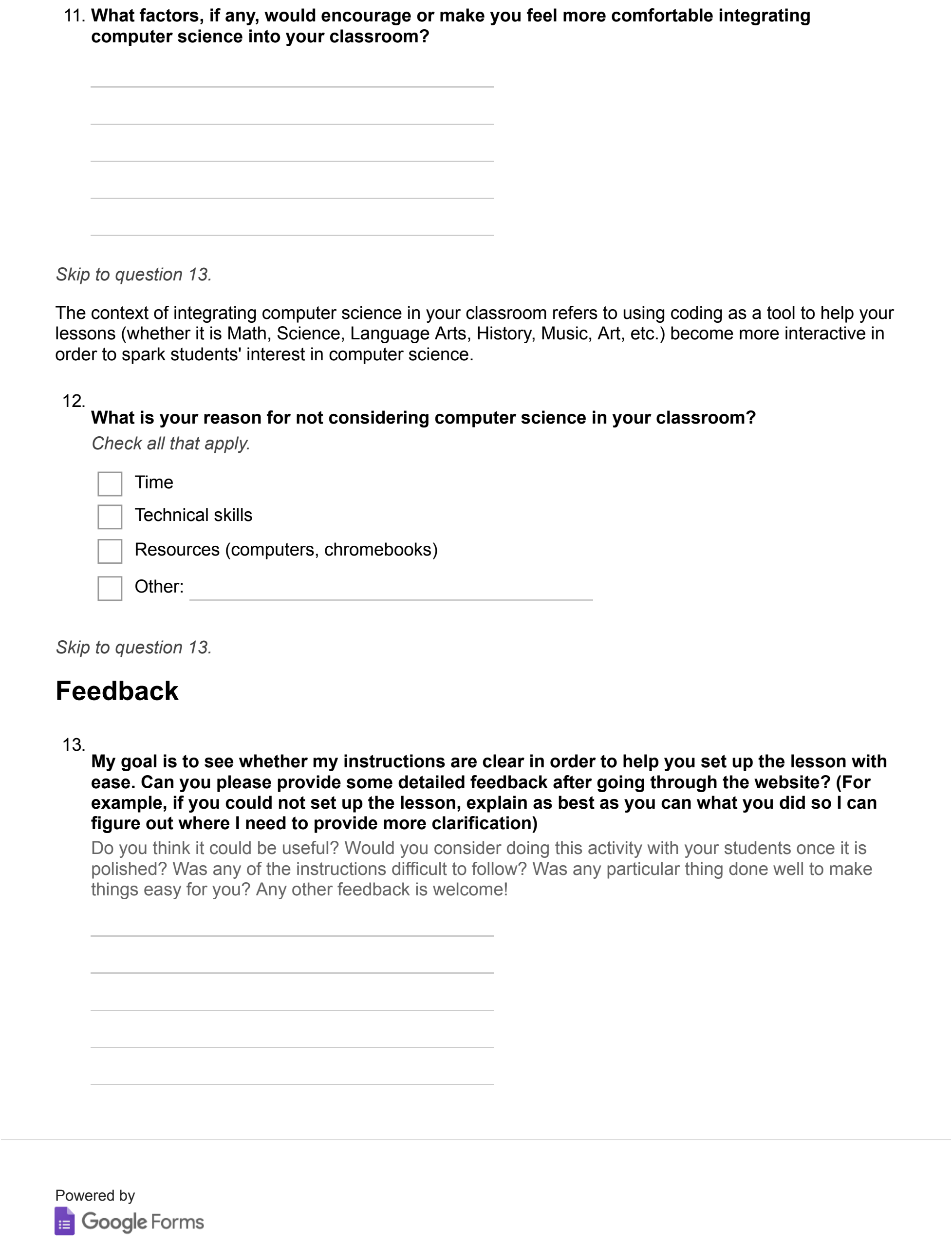
	1	2	3	4	5	
Not likely	○	○	○	○	○	Extremely likely

9. **How likely are you to integrate computer science with the subject(s) you teach in your classroom if someone came in to help you lead the lesson?**
Mark only one oval.

	1	2	3	4	5	
Not likely	○	○	○	○	○	Extremely likely

10. **How likely are you to integrate computer science with the subject(s) you teach in your classroom and lead the lesson yourself if you had a good tool to help you prepare?**
Mark only one oval.

	1	2	3	4	5	
Not likely	○	○	○	○	○	Extremely likely



11. **What factors, if any, would encourage or make you feel more comfortable integrating computer science into your classroom?**

____

____

____

____

____

Skip to question 13.

The context of integrating computer science in your classroom refers to using coding as a tool to help your lessons (whether it is Math, Science, Language Arts, History, Music, Art, etc.) become more interactive in order to spark students' interest in computer science.

12. **What is your reason for not considering computer science in your classroom?**

Check all that apply.

□ Time

□ Technical skills

□ Resources (computers, chromebooks)

□ Other: _______________________________________

Skip to question 13.

Feedback

13. **My goal is to see whether my instructions are clear in order to help you set up the lesson with ease. Can you please provide some detailed feedback after going through the website? (For example, if you could not set up the lesson, explain as best as you can what you did so I can figure out where I need to provide more clarification)**

Do you think it could be useful? Would you consider doing this activity with your students once it is polished? Was any of the instructions difficult to follow? Was any particular thing done well to make things easy for you? Any other feedback is welcome!

____

____

____

____

____

Powered by Google Forms