11-1-2017

Learning to Code Music: Development of a Supplemental Unit for High School Computer Science

Kelsey Wright
Boise State University

Learning to Code Music

Development of a Supplemental Unit for High School Computer Science

by

Kelsey Wright

A project

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in STEM Education (Computer Science Emphasis)

Boise State University

November 2017

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the project submitted by

Kelsey Wright

Project Title: *Learning to Code Music.* Development of a Supplemental Unit for High School Computer Science

Date of Final Oral Examination: November 2017

The following individuals read and discussed the project submitted by student Kelsey Wright, and they evaluated her presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Amit Jain, Ph.D.	Chair, Supervisory Committee
Jonathan Brendefur, Ph.D.	Member, Supervisory Committee
Tim Andersen, Ph.D.	Member, Supervisory Committee

The final reading approval of the project was granted by Amit Jain, Ph.D., Chair of the Supervisory Committee. The project was approved for the Graduate College by Tammi Vacha-Haase, Ph.D., Dean of the Graduate College.

TABLE OF CONTENTS

ABSTRACT

Learning to Code Music is a supplemental unit developed for high school computer science. This unit was developed after researching the effects of biases in curriculum, effective teaching, and incorporating the arts into coding. This supplemental unit is intended to be used with one of the Computer Science Principles curriculum approved by the College Board and explained in the literature review. It is my goal to have other teachers and myself to use this supplemental unit in their high school computer science courses. All supplemental unit material can be found at https://sites.google.com/notusschools.org/earsketch-csp/home

LIST OF FIGURES

LIST OF ABBREVIATIONS

APP	Application
STEM	Science Technology Engineering Mathematics
STEAM	Science Technology Engineering Art Mathematics
CSP	Computer Science Principles

CHAPTER ONE: LEARNING TO CODE MUSIC

Learning how to code within an application (app) has become a basic skill needed to negotiate today's world. Even something as intuitive and ubiquitous as a smart phone requires some knowledge and skill to set it up and configure its features to be able to use it for the reason it was purchased. The process of doing that is to provide data in the form of coded instructions that launch a set of apps and utilities that produce the configuration of the smart phone that the user needs. To the user, setting up, personalizing and configuring a smart phone is just making configuration choices. It would surprise the average smart phone user that they just coded their own phone.

My project is to develop a supplemental unit for a high school-level Computer Science course that introduces and teaches basic coding within an app. I created the course described in this paper as part of my studies this year (2016-2017). I have agreement for coordinating and collaborating with the Music teacher this school year (2017-2018) to create a cross-discipline Computer Science/Music opportunity that will pique interest in students who might not otherwise sign up for a coding class. My goal is that this unit will teach, surprise and inspire students to further explore coding and music among other arts in future classes.

The *Learning to Code Music* unit will allow any student to create their own music by coding the EarSketch app. In this unit, the student can focus on creating personal music without ever learning how to play a musical instrument. They will create their own music by learning specific coding fundamentals – instructions, syntax, etc. to code

the app. And, more important, the students will understand how these coding fundamentals actually come together to create a well-written, functioning program.

Specifically, the students will work through the EarSketch training material by writing pieces of code, instantly listen to the results through audio feedback, and learn to fix their errors by debugging the code they just created. Debugging is the process of identifying and removing errors that occur in programs. My supplemental material works with the EarSketch material to highlight the mechanics of coding and to structure supporting quizzes and projects to ensure they understand. I believe this is a much stronger approach (and more interesting) than just copying sample code to produce a simple result.

Expert programming requires the ability to abstractly define concepts, think logically, become very detail oriented, and be able to design, construct and test code against requirements using the fundamentals of mathematics and science. These are important abilities, but I have discovered in my studies and experience, that anyone can develop the skills to code within a structured application. The challenge for teachers who are focused on preparing students for a useful life in today's complicated, technology-available world is to change how we teach. The traditional methods of teaching do not work well anymore for most students. Students get very little experience in actually building things and in turn struggle to remember important problem solving concepts. Students need to see cross-discipline connections and especially connection to real world applications and that is what the STEM approach to teaching does work. It is even better

as STEAM to include the Arts, it reflects real life and better resonates with high school students.

I teach and have a passion for Math, but when I connected and finally understood the relationship between Math and Computer Science, I knew the Masters of Science in STEM Education with a Computer Science emphasis was the avenue I wanted to pursue.

CHAPTER TWO: LITERATURE REVIEW

The current goal for the field of Computer Science at the high school level is to increase students interested in the field and include more underrepresented culture. According to the U.S. Department of Education, in 2010 only 16 percent of high school students are interested in a STEM career and have proven a proficiency in mathematics. Currently, nearly 28 percent of high school freshman declare an interest in a STEM-related field, but 57 percent of these students will lose interest by the time they graduate from high school [U.S. Department of Education, 2017]. These numbers may have changed recently with the current push from colleges, STEM agencies, and high schools giving more opportunities to STEM education and especially computer science.

In preparation for creating this unit, I researched topics on incorporating the arts into STEM, curriculum bias, and effective teaching practices.

The Arts can serve as an on-ramp to Computer Science for underrepresented students. Incorporating the Arts into content areas, such as STEM, has been shown in numerous studies to improve long-term retention of content [Sousa, 2013].

Engaging students' strengths using art activities increases motivation and the probability of STEM success [Jolly, 2017]. For years, researchers have been reporting positive associations between music experience and cognitive growth in nonmusical areas among young children [Sousa, 2013]. This musical connection to a challenging topic like coding help introduce underrepresented students to the coding experience. Some students may also like and want to study drawing, so they can learn how to use code to create their own custom artwork. Even students with an interest in writing can write their own lyrics

for songs that they want to code. Using the approach of incorporating the Arts can create a gateway to computer science by taking advantage of creative expression while supporting underrepresented students' lack of exposures to computer science [Gaskin, 2016].

One concern when creating engaging and effective curriculum is biases. Biases can easily be written within a curriculum without the knowledge or without intent by the developer. Developing and following an anti-bias curriculum has the outcome of creating secure, respectful and reciprocal relationships that value diversity among people [Anti-Bias Curriculum, 2017].

In order for students to feel completely comfortable, especially in a new environment, they need to respect and embrace differences between individuals. This includes any and all biases among adolescents.

Teachers must also be cognizant of their own personal biases they might bring into their teaching and the classroom environment. Some possible biases in curriculum include invisibility, stereotyping, imbalance, unreality, fragmentation and isolation, and linguistic biases [Seven Forms of Bias, 2017].

Even if the curriculum is unbiased, a teacher needs to be aware of biases that they bring into their teaching or into their own creation of supplemental materials. Although any teacher may try to be neutral towards their students, we all have values, beliefs, and principles that impact how we teach our students on a daily basis. When our assumptions are manifested as implicit biases that can lead to imposed identities, we often reinforce negative stereotypes and negative school climates without even knowing it [Hanselman,

Bruch, Gamoran, & Borman, 2014]. Considering the racial and cultural demographic changes we see in the education system, teachers not only need to become prepared to meet the needs of this multicultural population of students coming into the public schools; they also need to increase their understanding of how as educators they have the potential to reproduce or interrupt oppressive cultural dynamic in the schools [Bersh, 2009]. The types of music chosen by students to create should be supported as personal expressions and not pre-selected based upon a teacher's personal bias. Music is "shared emotion" and "freedom of expression". Students should have the opportunity to express their own emotions and passions.

There are five Computer Science Principles curricula endorsed by College Board [College Board, 2017]. The EarSketch-based *Learning to Code with Music* supplemental unit that I created is to be in addition to one or more of these curricula.

These curricula include:

- Beauty and Joy of Computing
- Code.org
- Mobile Computer Science Principles
- Project Lead the Way
- UTeach CS Principles.

The Beauty and Joy of Computing curriculum [Beauty and Joy of Computing, 2017] focuses on computer science concepts with high level topics including recursion and higher order functions while utilizing Snap [Snap, 2017] as the visual coding

language. This curriculum consists of seven units including an introduction to Snap programming, conditionals and abstraction, lists, the internet and global impact, algorithms and data, threes and other fractals, and recursive and higher-order functions.

Code.org [Code.org, 2017] is a curriculum with six units including the internet, digital information, algorithms and programming, big data and privacy, building apps, and create and explore performance tasks. Code.org's curriculum utilizes JavaScript language using Code.org's App Lab environment to create small applications (apps).

Mobile Computer Science Principles [Mobile CSP, 2017] contains lesson plans, assessment materials, and other resources that utilize App Inventor [MIT App Inventor, 2017] as the visual coding language. Mobile CSP consists of nine units including setting up the App Inventor account, introduction to the visual coding language, creating graphics and images, animations, algorithms and procedural abstraction, using and analyzing data, communicating through the internet, Advanced Placement exam prep, and beyond the exam. The curriculum contains performance tasks and a midterm and final exam.

Project Lead The Way [Project Lead The Way, 2017] is a curriculum that includes four units algorithms, graphics, and graphical user interfaces, the internet, raining reigning data, and intelligent behavior. This curriculum is designed to fill 165 educational days with unplugged activities, projects, and uses python [Python, 2017] as a primary tool, but incorporates multiple platforms and languages such as Scratch [Scratch, 2017] and App Inventor [MIT App Inventor, 2017] for computation.

UTeach Computer Science Principles [UTeach CS Principles, 2017] is a year-long curriculum including online lesson plans, project based modules, projects, and utilizes Scratch [Scratch, 2017] and Processing [Processing, 2017] as their visual coding languages. UTeach contains seven units including computational thinking, programming through scratch, data representation, digital media processing through processing, data sets, innovative technology, and performance tasks.

These five curriculum vary in difficulty and depth of knowledge. Beauty and Joy of coding, Code.org, and Mobile CSP use language tools that work with blocks-based languages where students drag and drop premade code blocks. U-teach and Project lead the Way use both blocks-based and text-based programming which increases the difficulty of coding. The EarSketch and my supplemental material will fit in the text-based programming category and would pair nicely with the programs listed.

It is of note that many of the AP approved curriculum use platforms that utilize the drawing or animation side of the arts but none of them utilize the creation of music through code.

CHAPTER THREE: METHODOLOGY AND DESIGN

The unit I created utilized the curriculum from Earsketch in sections one through six and section nine. The reason I used only these sections is because sections one through six and nine give a good overview of using the Earsketch program while covering basic concepts of introductory coding. I wanted to create the unit to be supplemental and to give students enough time to really dive into coding music without losing their interest. This is why my unit only used seven of the twenty one sections available in the EarSketch curriculum. The implementation of this supplemental unit was designed to take two to three weeks for the students to complete. However, it can take more time if students are holding interest and teaching time in available. In my experience with high school students, each supplemental unit using some form of program or hardware (EarSketch, Processing, Arduino, Drones, etc.) will only remain interesting and keep students attention when covered over two to four weeks.

I wanted to develop my unit to be easy for teachers to follow and to keep student attention while creating fun and relevant projects. I designed each section project to have a different scheme. The purpose of this was to keep students interests as they move through the sections. This approach is a method that I discovered when researching anti-bias curriculum development. Giving students a variety of assessments, projects, and learning tools keeps from teaching to one type of student while respecting and embracing differences. Along with different methods of assessments, the projects were created with different approaches to reach more students and test different types of skills and ability levels. The first two projects are at a basic level of creating a song using what they have learned in the first two sections. In the third section students learn about commenting and

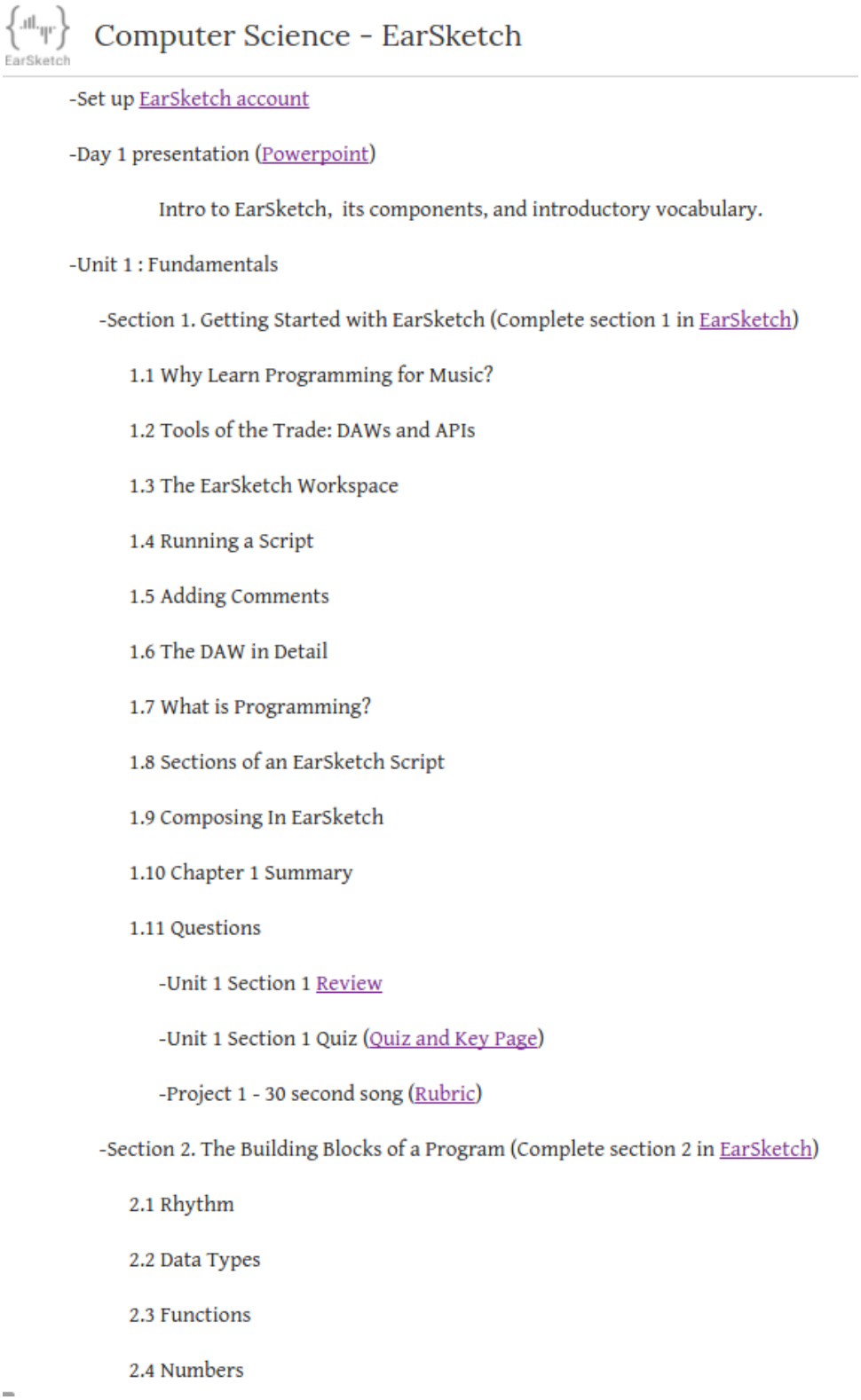
debugging. I felt that the best way to practice this challenging task is to do it themselves. That is why the third project is to comment and debug a song that has already been created with two different types of errors. There is a presentation along with this project to explain in more detail how to comment in both languages available to use in EarSketch. Then after the students debug the sample code the presentation walks through what they should have found and how they needed to fix the errors. The fourth project gives the students a different view on the process of creating songs by looking at the lyrics that go along with a song. There is a PowerPoint presentation that goes along with this project where students will learn about what it takes to write lyrics. They will then write their own lyrics and create the audio that would play along with those lyrics. I left recording the lyrics over the song as extra credit, because I wanted students to feel comfortable writing and creating this project. I know how many students get nervous when recording their own voice and I did not want that to interfere with the effectiveness and student creativity of this project. The fifth project is where the students create their own custom beat. This project also has a presentation that goes along with it to teach the students how to create a custom beat in EarSketch and gives a background on what a beat is and how it related to mathematical concepts. The final project has students create one last song using all of the concepts that they have learned throughout the selected sections. Each of these projects can be completed on an individual level or with pair programming based on the levels and comfort of the students. The assessments model different ways for students to express their knowledge through a variety of projects and section quizzes without over testing. In our public school system students are tested over and over throughout the school year. I wanted to create some assessments to give the

students accountability without taking the fun away from creating their own music. Prior to each section quiz is a section overview PowerPoint that highlights the concepts I would have liked each student to obtain through the sections. Giving students an overview helps them recap on the section and review to be successful on the section quizzes.

CHAPTER FOUR: NAVIGATING THE WEB PAGE

The way I designed the unit is how I would teach it to my computer science classes. I would cover most of unit one on the fundamentals and loops from unit two. This will give a good overview on how EarSketch works and it gives a foundation on creating music through code. The material that I created with an overview of the unit was developed and then put on a google site for other computer science teachers to access. The first page of the site is a description of the purpose of the sight and how I think it should be used, as shown in figure 1. Then the next page is the overview with links to all of the content I created. This overview page contains an outline to flows from top to bottom of the page, as shown in figure 2. The first task is for students to set up an EarSketch account and next is a day one PowerPoint with an overview of the EarSketch components as well as basic coding vocabulary that will be used throughout the unit, as shown in figure 3.

Figure 1. Home Screen of EarSketch web page.



Computer Science - EarSketch

-Set up EarSketch account

-Day 1 presentation (Powerpoint)

Intro to EarSketch, its components, and introductory vocabulary.

-Unit 1 : Fundamentals

-Section 1. Getting Started with EarSketch (Complete section 1 in EarSketch)

1.1 Why Learn Programming for Music?

1.2 Tools of the Trade: DAWs and APIs

1.3 The EarSketch Workspace

1.4 Running a Script

1.5 Adding Comments

1.6 The DAW in Detail

1.7 What is Programming?

1.8 Sections of an EarSketch Script

1.9 Composing In EarSketch

1.10 Chapter 1 Summary

1.11 Questions

-Unit 1 Section 1 Review

-Unit 1 Section 1 Quiz (Quiz and Key Page)

-Project 1 - 30 second song (Rubric)

-Section 2. The Building Blocks of a Program (Complete section 2 in EarSketch)

2.1 Rhythm

2.2 Data Types

2.3 Functions

2.4 Numbers

Figure 2. Overview with materials.

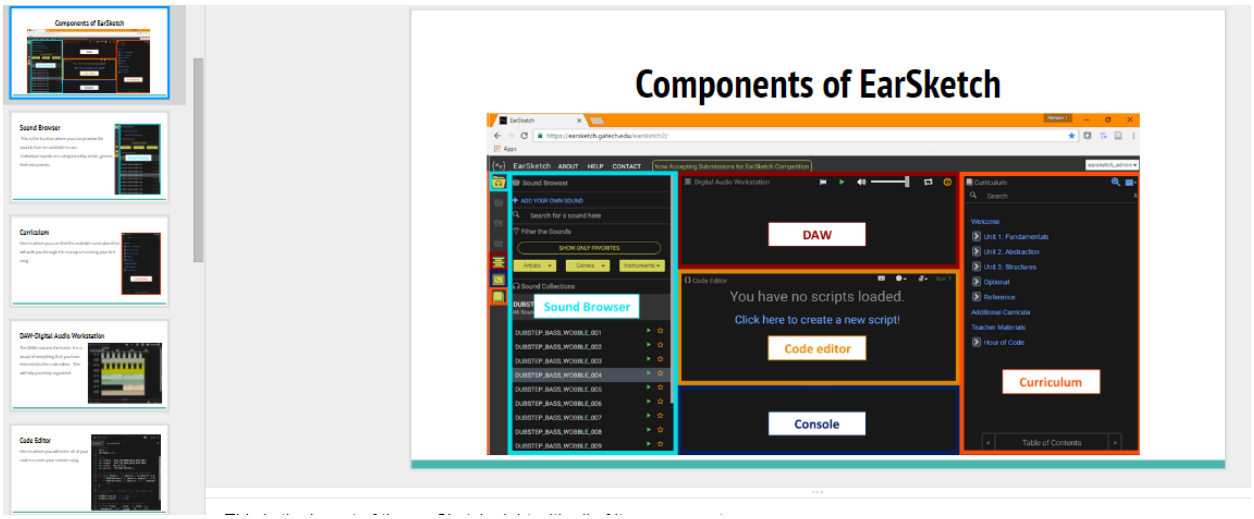


Figure 3. Day one presentation.

Each section is broken down in a similar style. The section design is visible back in figure two. First students will go through the section layout on the EarSketch website. They will complete the questions given by EarSketch and then move into the supplemental part of that section. This is where the class will go through a PowerPoint review of key concepts in that section, as shown in figure 4.

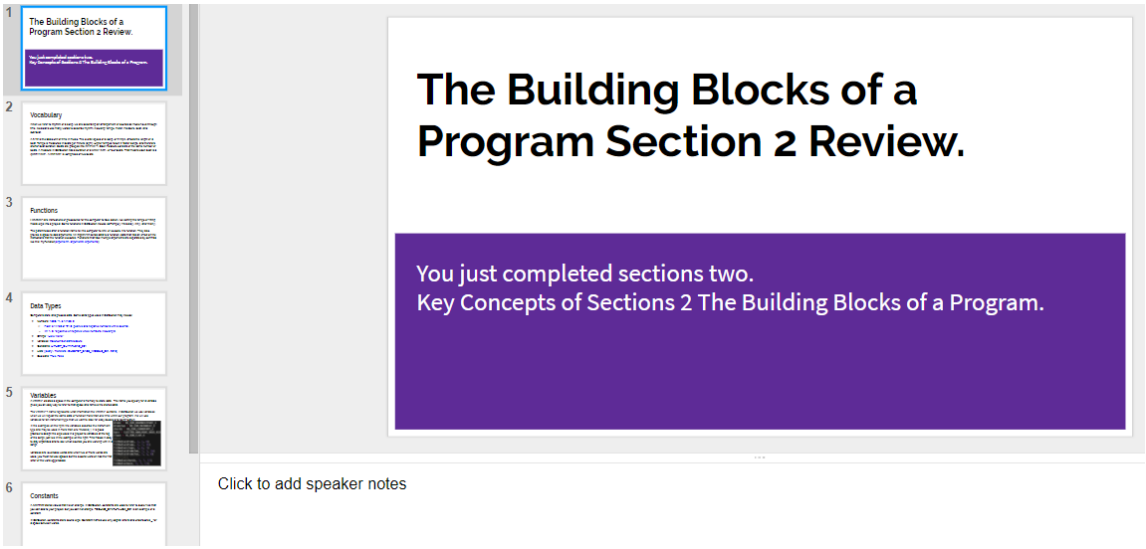


Figure 4: Display of a PowerPoint review.

Following the review, students will take a short quiz over the section. These quizzes and their keys are located on the third and final page of the site, as shown in figure 5. Each quiz has seven to twelve questions ranging from fill in the blank to multiple choice style questions. The final piece of the section is a mini project where students are able to model the level of knowledge that they have gained. Each project has an attached rubric to help guide students and assist teachers in grading student projects. The flow of each section was designed the same so students and teachers know what to expect, but the style of the project are different in order to keep the attention of the students and adhere to a variety of interests.

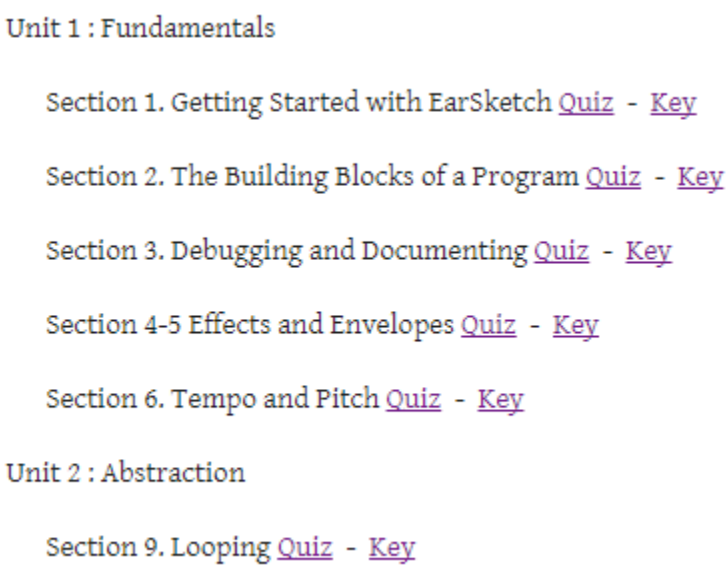


Figure 5: Quizzes and Keys Page

CHAPTER FIVE: CONCLUSION

I will be starting my third year of offering a Computer Science course at my school and recruiting students to select Computer Science is still a challenge. I want to offer an experience where students brag in the hall about what they are experiencing and creating. This helps get new students interested in the class for the next year and keeps the program going. The more students hear about this exciting class, the higher my enrollment will be.

I have begun the process of collaborating with our school's Music teacher and we are planning on playing some of the student created songs during the school's music concerts. These songs may be playing when families visit or we may even have some students play their music and talk about how it was created. I think it is very important to keep students interested in what they are working on and I feel EarSketch will do this.

Using my experience to give other teachers' advice on creating their own supplemental units, I would say to become aware of personal biases and other biases that can be unintentionally embedded in curricula. Through my research, I learned that some of these biases include invisibility, stereotyping, imbalance, unreality, fragmentation and isolation, and linguistic biases. Given these biases, someone creating their own supplemental curriculum should avoid underrepresentation of certain groups, assuming only traditional roles or attributes to a group, denying students' knowledge of the diversity and complexity of individuals, presenting only one interpretation of an issue, and ignoring viewpoints through selective presentation of materials. There may be biases in curriculum, teacher delivery, or through student interpretation. Being aware of such

biases and learning how to minimize them will help deliver content and engage more students in the subject. It will also help to find out what interest and inspire students and tie that into the unit. Do not be afraid to think outside the box. Add in a variety of different types of assessment methods. Make the material easy to access and follow.

Learning to Code Music can be more effective if teachers determine what it is they really want student to know and do as a result of this supplemental unit. Teachers will then be able to add or subtract content from rubrics, quizzes, and projects based on what they want student to know. Computer Science educators should think about and continually assess their personal understanding of effective teaching practices and biases in light of the evidence the classroom experience provides. Teachers should experiment with different teaching approaches and monitor the results. Some approaches may include pair programming versus individual programming. Students should be encouraged to assess their own learning as well as their understanding of how they learn, by giving them opportunities to reflect on the learning process. For researchers, the implication is that more studies need to be conducted on bias and using music to incorporate the Arts into STEM. The implications of Learning to Code Music for programming and computer science is that it adds one more tool for grabbing students interests and attracting them to the field of computer science and specifically to programming. Computer Science has had a big impact on the growth of modern society. In today's world utilizing powerful tools like EarSketch can help bridge the gap in industry need and student interest in Computer Science. Implications for mathematics is that more research needs to be done in the connection between mathematical patterns and concepts and their relation to Earsketch and coding music.

REFERENCES

Anti-Bias Curriculum in Early Care and Education. Bicultural Inclusion Support Services. imagineeducation.com . Accessed 8 Mar. 2017.

Anti-Bias Education. Teaching for Change. 2017. teachingforchange.org. Accessed 8 Mar. 2017.

Beauty and Joy of Computing. University of California, Berkeley and Education Development Center, Inc. bjc.edc.org. Accessed 9 Mar. 2017.

Carime Bersh, L. Deconstructing Whiteness*: Uncovering prospective teachers' understandings of their culture-A Latina professor's perspective.* Multicultural Perspectives. 2009.

College Board. collegeboard.org. Accessed 8 Mar. 2017.

EarSketch. Georgia Tech Research Corporation. earsketch.gatech.edu. Accessed 8 Mar. 2017.

Fry, Ben and Casey Reas. *Processing*. processing.org. Accessed 8 Mar. 2017.

Gaskins, Nettrice. *Making Art and Dance Are Making Computer Science Culturally Relevant.*

EdSurge News. 2016. edsurge.com. Accessed 8 Mar. 2017.

Hanselman, P., Bruch, S. K., Gamoran, A., & Borman, G. D. *Threat in Context: School Moderation of the Impact of Social Identity Threat on Racial/Ethnic Achievement Gaps.* Sociology Of Education. 2014

Jolly, Anne. *STEM vs. STEAM: Do the arts belong?* Education Week Teacher, 2017.

Library of Congress. congress.gov. Accessed on 8 Mar. 2017.

MIT App Inventor. appinventor.mit.edu. Accessed on 9 Mar. 2017.

Mobile CSP. mobile-csp.org. Accessed on 9 Mar. 2017.

Project Lead The Way. pltw.org. Accessed on 9 Mar. 2017.

Python. python.org. Accessed on 9 Mar. 2017.

Robinson-Cimpian, J. P., Lubienski, S. T., Ganley, C. M., & Copur-Gencturk, Y. *Teachers' perceptions of students' mathematics proficiency may exacerbate early gender gaps in achievement.* Developmental Psychology. 2014.

Rosenshing, Barak. *Principles of Instruction. Research-Based Strategies That All Teachers Should Know*. American Educator, 2012. aft.org. Accessed on 13 Mar. 2017.

Science, Technology, Engineering and Math: Education for Global Leadership. Department of Education. ed.gov. Accessed on 9 Mar. 2017.

Scratch. scratch.mit.edu.. Accessed on 13 Mar. 2017.

Seven Forms of Bias. Pennsylvania Department of Education. education.pa.gov. Accessed on 15 Mar. 2017

Snap. snap.berkeley.edu. Accessed on 9 Mar. 2017.

Sousa, David and Tom Pilecki. *From STEM to STEAM*. Corwin. 2013.

U.S. Department of Education. ed.gov. Accessed on 5 Mar. 2017.

UTeach CS Principles. National Science Foundation. cs.uteach.utexas.edu. Accessed on 13 Mar. 2017