**SECONDARY COURSE DESCRIPTION**

**SECTION A: COURSE CLASSIFICATION**

ALAMEDA UNIFIED SCHOOL DISTRICT
Excellence & Equity For All Students

| | |
|---|---|
| 1. Course Title:<br>AP Computer Science Principles (CTE) | 6. Prerequisite(s):<br>Completion of one other computer science pathway course |
| 2. Action:<br>_x__ New Course<br>___ Course Revision<br>___ Title Change Only | 7. Grade Level:<br>10-12 |
| 3. Transcript Title/Abbreviation:<br>AP ComScPrCTE<br>(For Educational Services) | 8. Elective/Required:<br>Elective |
| 4. Transcript Course Code/Course Number:<br>(For Educational Services) VQPM | 9. Subject Area:<br>Elective |
| 5. CBEDS Code:<br>(For Educational Services) 8132 OR 2472 | 10. Department:<br>CTE |

11. Length /Credits:
___ 0.5 (half year or semester equivalent) _x__ 1.0 (one year equivalent)
___ 2.0 (two year equivalent)

12. Was this course previously approved by UC? ___ Yes _x__ No
Will be submitted for Approval

13. Meets the "_____" requirements in the a-g university/college entrance requirement. Approval date:
_____-

14. School Contact Information
Name: _____Nancy Read_____ Title/Position: _____Teacher_____
Phone: _____510.337.7022_____ Fax: _____
E-Mail: _____nread@alamedaunified.org_____

16. Signatures:
Department Chair: _____
Principal: _____
Acknowledged by Other Principals: _____
_____
_____
Educational Services: _____
_____

16. BOE Approval
Signature of Superintendent:_____ Date of Approval_____

**SECTION B. COURSE CONTENT**

17. Course Description:

This year-long course is the second course in the Career Technical Education ICT/Computer Science Pathway. The course will cover the basic principles of computer science (CS) from the perspective of mobile computing. Through block coding, students will create socially responsible mobile apps on the Android platform while learning the computer science big ideas: creativity, abstraction, data and information, algorithms, the internet, and global impact. Speakers from the IT industry and field trips to see high tech companies will help students learn about career opportunities. Students gain entry-level skills that will help them to secure internships and to help determine if computer science is a field they want to pursue in college. Students will be given extra assignments that will prepare them to take the AP Computer Science Principles Exam. Students who successfully complete this course will also be prepared for AP Computer Science A (java).

18. Course Goals and/or Major Student Outcomes: (see outline)
19. Course Objectives (standards): (see outline)
20. Course Outline:

## Outline of Curriculum Units and Projects:

The units that follow interweave the six AP CS Principles Computational Thinking Practices of Connecting Computing, Creating Computational Artifacts, Abstracting, Analyzing Problems and Artifacts, Communicating, and Collaborating with the seven CS Principles Big Ideas of Creativity, Abstraction, Data, Algorithms, Programming, Internet, and Global Impact.

- *Unit 1 - Getting Started: Preview & Set up*
- *Unit 2 - Introduction to Mobile Apps & Pair Programming*
- *Unit 3 - Creating Graphics & Images Bit by Bit*
- *Create - Programming Performance Task #1 (Practice)*
- *Unit 4 - Exploring Computing: Animation, Simulation, & Modeling*
- *Exam #1*
- *Explore - Impact of Computing Innovations Performance Task #1 (Practice)*
- *Unit 5 - Algorithms & Procedural Abstraction*
- *Explore - Impact of Computing Innovations Performance Task #2*
- *Unit 6 - Using and Analyzing Data & Information*
- *Unit 7 - Communication Through the Internet*
- *Create - Programming Performance Task #2*
- *Final Portfolio*

### Unit 1: *Getting Started: Preview and Set up (Creativity, Algorithms, & Impact)*

Unit 1 of the course provides a brief overview of the curriculum, emphasizing its main theme: learning the principles of computer science while building socially useful mobile apps. The hands-on work focuses on setting up the student's digital learning environment, including their integrated programming environment, online portfolios, and teacher chosen learning management tools (e.g. Google Classroom and Aeries). Students learn basic digital citizenship, compute skills, and lab expectations before a brief introduction to the course curriculum. Students try blocks-based programming by working through a series of increasingly challenging Blockly Maze problems. Additionally, a brief introduction to the reading resource sand work expectations used throughout the course are introduced. Finally, since the course and ICT

field both require teamwork, students are introduced to and practice group dynamics including leadership, collaboration, and communication skills.

**Guiding Questions:**

- What is the CS Principles course?
- What is graphical blocks-based programming?
- Why is it important to study the impact of computing technology?

| **Lessons:**<br>Welcome to CSP,<br>**Mazes, Algorithms, and Programs,**<br>Google Account and Portfolio Setup,<br>App Inventor Setup,<br>Blown to Bits (BB),<br>Joining the Forum,<br>Wrap up | **Instructional Activity: Mazes, Algorithms, and Programs**<br><br>The purpose of this activity is to show an example of what blocks-based programming is like and to introduce some basic terminology. Students work individually or in pairs to complete a sample blocks-based activity to create small programs (*scripts),* to solve mazes. This activity focuses on algorithm and programming concepts.<br><br>**CTE Standards:**<br>Communication: 2.5; Problem Solving: 5.5, 5.8; Responsibility: 7.3; Leadership & Teamwork: 9.2, 9.5-9.7; Technology: 10.5, 10.10, 10.11; Develop software using programming languages: C4.3, C4.6; Develop Web and online projects C7.1, C7.5 |
|---|---|
| **Labs:** Mazes, Algorithms, and Programs (Blockly), App Inventor Setup (App Inventor); Setup Google Classroom, Aeries, and Digital Portfolios | |

## Unit 2: *Introduction to Mobile Apps and Pair Programming (Creativity, Abstraction, Programming, & Impact)*

Unit 2 introduces App Inventor 2, or the chosen programming platform, and the course's first programming project, a basic soundboard app. Students use App Inventor's ***event-driven programming*** model to work through a guided tutorial that plays an excerpt of a Martin Luther King speech. Next, they work through several *exercises* that challenge them to extend their understanding by solving problems on their own. Later in the unit working in pairs, students extend their learning by incorporating *creative mini projects.* In the final stage of the app development, students express themselves by incorporating their own ideas in a personal soundboard *computational artifact*. Additionally, several important CS Principles themes and topics are covered including ***hardware*** and ***software*** concepts and the big idea of ***abstraction.*** Students also get their first look at ***binary numbers*** learning how to count in binary and how to view number systems such as binary, hexadecimal and decimal, as instances of the higher order abstraction of a *positional number system*.

**Guiding Questions:**

- How does one use App Inventor and event-driven programming to build a mobile app?
- What are the various hardware and software abstractions that make up a modern digital computer?
- What is the binary number system that underlies all digital representation?

| **Programming Lessons:** | **Instructional Activity: Soundboard Projects** |
|---|---|
| Soundboard Tutorial parts 1 and 2,<br>**Soundboard Projects**<br><br>**Concept Lessons:**<br>What is Abstraction?<br>Mobile Apps and Mobile Devices,<br>Binary Numbers,<br>Hardware and Software Abstractions<br><br>Required Reading: The Digital Explosion, | The ***Soundboard Projects*** lesson is the third and culmination of a series of three related lessons: students express their own ideas and implement enhancements and extensions to the app they have been studying. In the first lesson students follow an instructor-led tutorial on how to build a basic soundboard app (I Have a Dream). The instructor introduces basic App Inventor programming concepts, including the event-driven programming model used throughout the course. In the second lesson, students complete small but increasingly challenging exercises and encouraged to work ***collaboratively*** to figure out solutions on their own. In the culminating lesson, students design and implement enhancements and extensions to the app, including, possibly, creating an entirely new *soundboard app* based on their personal ideas and interests. These activities focus on creativity, abstraction, and programming concepts.<br><br>**CTE Standards:**<br>Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8, 5.10, 5.11; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Develop software using programming languages: C4.3, C4.5, C4.6; Integrate a variety of media into development projects: C6.1, C6.6, C6.7 |
| **Labs:** Soundboard Tutorial (App Inventor), Soundboard 1 (App Inventor), Soundboard Projects (App Inventor) | |

## Unit 3: *Creating Graphics & Images Bit by Bit (Creativity, Abstraction, Data and Information, Programming, & Impact)*

Unit 3 extends the student's mobile programming toolkit to several new App Inventor components and introduces several new programming concepts, including the concept of a *variables, lists* and *data abstraction*. The main app in this unit, The *Paint Pot* app, a computational version of finger painting, focuses on App Inventor's drawing and painting features and related topics from the CS Principles framework. The app has four parts each of which is followed by a set of creative project exercises and challenges. This unit also introduces another app: *Map Tour,* which demonstrates how to incorporate external data into a mobile app and have it persist in a data base. Unit 3 also extends the student's understanding of *binary number system* and introduces students to the idea of a *bit* as the fundamental unit of data. Through several hands-on and interactive activities students explore how bits are used to represent images, and how redundant parity bits can be used to detect simple data transmission errors. A reading that focuses on digital documents, including how information can be hidden inside images and other digital documents complements these lessons nicely.

**Guiding Questions:**

- How can binary numbers be used to represent all digital data?
- How can algorithms be used to compress data?
- How do variables of both simple and structured data, such as, lists, enable us to manage the complexity of a programming?
- How can you use a database to store information for the next time you use the app?

| **Programming Lessons:**<br><br>Paint Pot 1 (A finger painting app), Paint Pot 1 Projects, Paint Pot 2 (An introduction to variables, refactoring, and documentation), Paint Pot 2 Projects,<br>Map Tour Tutorial and Projects,<br><br>Conceptual Lessons:<br>Error Detection, Parity Error Detection,<br><br>Required Reading and discussion: Representing Images & electronic documents | **Programming Activity:**<br>Students work through their app activities as in the previous units. The Paint Pot app introduces students to the concept of variables. Students also learn how to make code easy to read by refactoring and the use of comments in their code. Map Tour shows students how databases can be used to make data persist once the app is turned off as well as an API in the form of Google Maps<br><br>**Instructional Activity: Representing Images**<br>Building on the student's knowledge of binary and hexadecimal number systems from the previous unit, students complete an activity that allows them to gain insight and knowledge of how binary numbers are used to represent all types of data, including numbers, images, characters, and machine language instructions. Students learn about *lossy* and *lossless* compression algorithms and EU 2.1 as students complete an unplugged (grid paper and pencil) activity in which they apply the *run--length encoding* algorithm to represent simple images in terms of numbers.<br><br>**CTE Standards:**<br>Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8, 5.10, 5.11; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6; Develop database: C8.1, C8.5-7 |
|---|---|
| **Labs:** Paint Pot 1 (App Inventor), Paint Pot 1 Projects (App Inventor), Paint Pot 2 (App Inventor), Paint Pot 2 Projects (App Inventor), Map Tour (App Inventor and Google Maps Activity Starter) | |

## Create: Programming Performance Task #1 *(Creativity, Abstraction, Algorithms, & Programming)*

Up until this point, students have completed App Inventor tutorials and app creation challenges. Students have 12-15 hours of class time to complete this task.

| **Assessment: Create Your Own Mobile App** |
|---|
| Students work ***collaboratively*** with a partner *(pair programming)* to create a socially useful, interactive, mobile app. The app must in some way include drawing, graphics, and programming constructs based on skills learned in prior lessons. Students learn how to brainstorm their ideas and develop wireframes with storyboards to express those ideas. Students give a 1-2-minute elevator pitch of their app idea and receive feedback from the instructor and their classmates. Students work during class time to develop, test, and debug their app. The instructor answers any questions and provides feedback along the way. While working on their app, students maintain a portfolio write up of their work making note of their progress and any challenges they may have faced, as well as screenshots of blocks of code with written explanations of the how the code works. Students record a video of their app. The project ends with |

an in-class presentation and app demo by each pair of students. This assessment and its activities focus on creativity, abstraction, algorithm, and programming concepts.

**CTE Standards:**
Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8, 5.10, 5.11; Integrate a variety of media into development projects: C6.1, C6.4, C6.6, C6.7; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6; Develop database: C8.1, C8.5-7

## Unit 4: *Animation, Simulation, and Modeling: Exploring the Impact of Computing (Creativity, Abstraction, Data and Information, Algorithms, Programming, & Impact)*

Unit 4 focuses on *animation, simulation and modeling*. The *Lights Out* app introduces the idea of *computer simulation* with a computational version of the traditional Whack-a-Mole game. The *Coin Flip* app, which extends over several lessons, introduces the concept of *modeling.* Students learn that models use abstractions, such as a pseudo random number generator (PRNG), to represent real world situations, in this case, the flipping of a coin. Students learn how PRNG algorithms are used to model *randomness* inside a computer, such as with the *Coin Flip* App. Students extend the app model to represent different types of coins, including a biased coin and a three--sided coin. Students experiment with their app to test and assess the quality of App Inventor's PRNG. Students learn how developing technology and computing innovations impact their privacy. Students learn the economic, social and cultural effects of computing innovations, such as real-world models of the weather and the solar system.

**Guiding Questions:**

- How do computers use simulation and modeling to represent real world phenomena?
- Why is randomness important and how is it modeled inside a computer?
- In what ways does simulation and modeling extend our knowledge and benefit society?

| **Programming Lessons:**<br><br>Lights Off Tutorial,<br>Lights off Projects,<br>Coin Flip Simulation,<br>Coin Flip Experiment,<br>Coin Flip Simulation Projects,<br>Logo, Part 1<br><br>**Conceptual Lessons:**<br><br>Pseudo Random Number Generators (PRNGs),<br>Abstraction: Inside the CPU<br><br>Required Reading:<br>Privacy | **Programming Activity:**<br>Students work through their app activities as in the previous units. The Lights Out app introduces the concepts of animation, timing, and randomness. The student will also see again the concept of a programmer-defined procedure, an important abstraction. Logo reinforces the use of procedures as abstractions as well as provides with practice in writing pseudocode. The Coin Flip app, which simulates flipping a coin, has students work **collaboratively** to conduct an experiment using a mobile app that models a coin flip. This experiment tests the hypothesis that App Inventor's PRNG is a good model of random behavior. Students record data in a table and calculate the percentage of heads and tails, which, in a good model should approach 50:50. Students work in small groups communicating their results to the class.<br><br>**Conceptual Activities:**<br>*Inside the CPU* leads students from a low-level programming language to a higher-level language in order to illustrate the layers of abstraction at work in a computer. It also illustrates some of the basic concepts around how hardware components work in the computer.<br><br>*Required Reading:* Students communicate their ideas and discuss how one's privacy is impacted by the developing technology and computing innovations<br><br>**CTE Standards:**<br>Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8, 5.10, 5.11; Technical Knowledge and Skills: 10.5;Integrate a variety of media into development projects: C6.1, C6.4, C6.6, C6.7; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Define and analyze systems and software requirement: C2.2; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6 |
|---|---|

**Labs:** Lights Out (App Inventor), Lights Out (App Inventor), Coin Flip (App Inventor), Coin Flip Projects (App Inventor), Logo (App Inventor)

**Explore: Impact of a Computing Innovation Performance Task #1** ***(Creativity, Impact)***

Up until this point, students have read various texts about recent computing innovations that have been in the news. Students are encouraged to find daily news articles about advances in technology and share them with the class. Students are given 4-5 hours of class time to complete this activity.

| **Assessment: Impact of a computing innovation** |
| --- |
| This assessment involves discussing, as a class, a computing innovation that has had considerable impact on the social, economic, or cultural areas of our lives, such as phone monitoring software. Students work **collaboratively** in small groups to research the computing innovation and find *reliable sources* using sites such as the *ACM Digital Library.* Students cite their sources and learn about *plagiarism.* The instructor assigns each group member a prompt taken from the official APCSP Explore Performance Task to answer about the innovation. Each group member answers the prompts in a single Google document shared among the group. The group then works together to edit the entire document discussing needed changes. When the document is completed (i.e. all prompts are answered and all sources are cited), each student is asked prepare their own original digital artifact (e.g. music, image, video, infographic, presentation, program, web page) to express the effects of the chosen innovation. Students share their artifact with their group members. |

This assessment focuses on creativity, data, and global impact concepts.

**CTE Standards:**
Communication: 2.4-7; 4.0 Technology: Problem Solving & Critical Thinking: 5.1, 5.4-6; Responsibility: 7.3-4; Leadership & Teamwork:9.2, 9.7 Technical Knowledge and Skills: 10.2

**Unit 5: Algorithms and Procedural Abstraction** ***(Abstraction, Algorithms, Programming, & Impact)***

In Unit 5, algorithms and procedures are examined in more detail. The Logo apps introduce the concept of procedural abstraction and students learn to define and use procedures -- named blocks of code that perform a specific task. By encapsulating the algorithms into named procedures and introducing parameters to help generalize the algorithms, students are led to see the advantages of procedural abstraction. In addition to designing and testing their own algorithms, students are also provided an introduction into the ***analysis of algorithms.*** Algorithm efficiency is examined for searching and sorting algorithms, which are analyzed both experimentally and through mathematical concepts such as functions and graphs. The impact section of this unit focuses on the impact that Web searching algorithms have had on our lives. The activities completed in Unit 5 build toward EU 2.2, EU 4.1, EU 4.2, EU 5.3 and EU 5.5 by focusing on abstraction, algorithms, and programming concepts.

**Guiding Questions:**

- How are multiple levels of abstraction used to create computational artifacts?
- In what ways are some algorithms better than others?
- What limits do algorithms have?

| **Programming Lessons:**<br>Logo Part 2, Quiz App, Quiz App Projects<br><br>**Conceptual Lessons:**<br>Analyzing Algorithms<br>Search Algorithms,<br>Sort Algorithms,<br>Limits of Algorithms<br><br>Required Reading:<br>Web Searches | **Programming Activities:**<br>**Logo 2**: In its initial version, students had very impoverished procedures to work with -- i.e., a *move* procedure that only moves the Android by 10 pixels and a *turn* procedure that only turns right by 90°. In this version, students use procedures with *parameters* as a more powerful abstraction. In this way, the Android can be made to move and turn by arbitrary amounts -- i.e., *move(x)* and *turn(y).* Students are then encouraged to develop their own procedures -- their own abstractions -- to draw more complex shapes. By adding simple loops into the procedures, students can design interesting graphical figures, which demonstrate the close interplay between algorithms and procedures. **Quiz App** focuses on storing data in parallel lists, where the data at each index is related. Students create a basic quiz app with lists of pictures, questions, and answers. The programming portion of the app emphasizes iterating over a list, where the user controls moving to the next item in the list, including how to handle reaching the end of the list. This lesson reinforces the enduring understanding that data is processed to gain insight and knowledge.<br><br>**Conceptual Activity: Limits of Algorithms**<br>In this lesson students use apps **collaboratively** to *classify* algorithms experimentally as either *logarithmic, linear, n log n,* or *quadratic.* A video introduces the concepts of *intractability* and *undecidability* through examples of (intractable) problems that cannot be solved efficiently and (unsolvable) problems that cannot be solved at all by means of an algorithm.<br><br>**CTE Standards:**<br>Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8, 5.10, 5.11; Integrate a variety of media into development projects: C6.1, C6.4, C6.6, C6.7; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6; |
|---|---|
| **Labs:** Logo 1 (App Inventor), Logo 2 (App Inventor), The Pong Game (App Inventor), Debugging Pong (App Inventor) | |

## Unit 6: *Using and Analyzing Data and Information (Creativity, Data and Information, Programming, & Impact)*

Unit 6 focuses on various aspects of using and manipulating ***Data***, both within mobile apps and on the Web and Internet. The App Inventor lessons in this unit focus on different types of programming data, including variables and ***structured data***, such as lists and databases. Students build apps that involve ***persistent data,*** data that persists from one instance of the app to another and learn how to share data online by using simple Application Programming Interfaces (APIs), such as the Google Fusion table API. This unit's CS Principles lessons focus on the concept of Big Data and its growing importance and its impact on society. Students are also introduced to the some of the algorithms for processing massive datasets.

**Guiding Questions:**

- How does continuous access to large amounts of data change how people and organizations make decisions?
- How do computers put things in order and find things in a list?
- What is the connection between data, information, knowledge, and wisdom?

| | |
|---|---|
| **Programming Lessons:**<br>Clicker App with TinyWebDB<br>Clicker App with Firebase<br><br>**Conceptual Lessons:**<br>Big Data<br>Visualizing Data<br>Visualizing Data Project<br><br>Required Reading:<br>Blown to Bits: Who Owns the Bits?, | **Programming Lessons:**<br>Students create a clicker App to respond thumbs up or thumbs down to a question. TinyWebDB, a persistent database across devices is used to keep track of the responses. Students then create a similar app using Firebase and compare and contrast the strengths and weaknesses of both.<br><br>**Conceptual Lessons:**<br>Students investigate the history of data storage, data sizes, processing large data sets, and the use of data to support innovation in other fields. Students develop an understanding of how computing enables discovery of connections in information. Students work in pairs to explore a data set of their own choosing. Students are required to develop hypotheses about their data and then test them by using Google Spreadsheets and Google My Maps to create visualizations.<br><br>**CTE Standards:**<br>Communication: 2.3; Problem Solving & Critical Thinking: 5.1, 5.4, 5.5, 5.8; Responsibility: 7.3, 7.4; Leadership & Teamwork: 9.7; Technical Knowledge and Skills: 10.6, 10.10; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6; |
| **Labs:** Clicker App with TinyWebDB (App Inventor), Clicker App with Firebase (App Inventor), Data Persistence Projects (App Inventor) | |

## Unit 7: *Communication Through the Internet (Creativity, Programming, The Internet, & Impact)*

Unit 7 focuses on the ***Internet***, one of the big ideas in computer science. The App Inventor lessons in this unit show different ways to use the internet in apps, including the ability to send text messages over Wi-Fi, finding directions via the Google Maps API. The CS Principles lessons focus on the Internet, how it works, how it enables innovation and collaboration, and security concerns for using it.

**Guiding Questions:**

- What is the Internet, how is it built, and how does it function?
- What aspects of the Internet's design and development have helped it scale and flourish?
- How is cybersecurity impacting the ever-increasing number of Internet users?

| Lessons: | Instructional Activity #1: How the Internet Works |
|---|---|
| What is the Internet?<br>No Texting While Busy Tutorial,<br>Cloud Computing and Ethics,<br>**How the Internet Works,**<br>My Directions Tutorial,<br>Cryptography Basics,<br>**Cryptography: Securing the Internet**<br>Blown to Bits: Cryptography,<br>Socially Aware App: Broadcast Hub,<br>Wrap up | This lesson goes more deeply into the infrastructure and mechanics of the Internet. It explains *packet switching*, *TCP/IP* and the protocol hierarchy. Students complete a series of activities using network administration software tools such as *Ping* and *Traceroute*, as well as, looking up *domain names*, and *IP addresses.*<br><br>**CTE Standards:** Communication: 2.3, 2.5, 2.7; Technology: 4.3; Problem Solving and Critical Thinking: 5.2, 5.4, 5.5, 5.6; Technical Knowledge and Skills: 10.5, 10.13; Develop software using programming languages: C4.3-7 4.11; Test, debug, and improve software development work: C5.3-4, C5.6<br>**Instructional Activity #2: Cryptography: Securing the Internet**<br><br>After learning about and using some of the basic concepts of cryptography in an earlier lesson, students are introduced through ***CS Unplugged*** videos to the key-exchange problem and then to the basic ideas of public-key encryption as a way of solving this problem. Through a video lecture, students learn essential the details about the Internet's trust system and how it is implemented in modern browsers to support the exchange of information securely across the Internet.<br><br>**CTE Standards:** Communication: 2.3, 2.5, 2.7; Technology: 4.3; Problem Solving and Critical Thinking: 5.2, 5.4, 5.5, 5.6; Technical Knowledge and Skills: 10.5, 10.8 |
| **Labs:** No Texting While Busy (App Inventor), My Directions (App Inventor and Google Maps APIs), Broadcast Hub (App Inventor) | |

21. Instructional Materials:
    Board approved required text:

    Supplementary materials:

App Inventor 2: Create Your Own Android Apps. David Wolber, Hal Abelson, Ellen Spertus, and Liz Looney O'Reilly Media, Inc., 2014

Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion. Hal Abelson, Ken Ledeen, Harry Lewis. Addison-Wesley, 2010

22. Instructional Methods and/or Strategies

**Programming Environment:**

App Inventor for Android (ai2.appinventor.mit.edu), a free online software platform, is used in this course to build mobile apps for Android devices.

**Online Resources:**

The ***complete curriculum*** is hosted online and free of charge: https://ram8647.appspot.com/mobileCSP. The course uses many freely available resources that are only available online to ensure that the course material is current and adaptable. Students maintain individual online portfolios of their course work by using Google sites (https://www.google.com/sites/overview.html). Self-check and live coding exercises make use of Quizly (https://github.com/ram8647/quizly), a Web-based live coding platform for App Inventor. Throughout the course, students will also use a number of online articles and videos from sources such as The New York Times (www.nytimes.com), Wikipedia (www.wikipedia.org), CS Bits and Bytes (http://www.nsf.gov/cise/csbytes/), Logic.ly (www.logic.ly), YouTube (www.youtube.com), and
CS Unplugged (http://csunplugged.org).

23. Assessment and Evaluation

## Assessments:

### Portfolios

In this course, students will document their work on personal digital portfolios. This work will include resumes, work samples from other classes, as well as written responses to assigned readings, documentation of creative programming projects, and their advanced create and explore performance tasks. The portfolios will promote collaboration and sharing of ideas and will constitute a full record of what the students have done in the course. Portfolios are also used as a summative assessment tool throughout the duration of the course.

### Reading and Homework Assignments

There will be regular reading and/or out-of-class homework assignments. These may include reading a chapter from the textbook and/or completing a tutorial or worksheet.

### Labs

This course takes place in a computer lab. Students have access to computers and mobile devices and any other necessary hardware, both during the class and during free periods. Internet access is available to students throughout the course. In each unit, there are at least three labs designed to practice and/or reinforce key concepts. Some are unplugged and others completed in an online Integrated Development Environment (IDE).

### Projects

There will be one (1) creative programming project in which students will use lab time to work either individually or collaboratively (in pairs) to create a socially useful mobile app that they propose (pitch), design, and implement. This project is based on the official College Board Create Performance Task for AP CSP.

There will also one (1) written research project that students will work on either individually or collaboratively. This research project will focus on examining a computing innovation that has a social impact on society. This project is based on the official College Board Explore Performance Task for AP CSP.

### Oral and Video Presentations

As part of the create task, students will prepare a video of their app and development process. In addition to presenting their Create and Explore tasks, students will have other opportunities throughout the course to make class to build and reinforce their communication and collaboration skills.

### Quizzes and Exams

There will be periodic quizzes to check students for understanding of key concepts and vocabulary throughout each unit. Quizzes will be hand written and/or electronic and exams will be electronic. Midterm and Final Exams will be project based and have a presentation component to them. The Midterm and Final projects can be individual or collaborative.

**Self-Check and Live Coding Exercises**

Assessments are an essential part of the learning process. Short, interactive, self-check exercises that consist of multiple choice and fill-in question as well as automatically graded live-coding programming exercises (https://github.com/ram8647/quizly) accompany all lessons in this course. Each question or exercise includes detailed feedback and students may repeat the question or exercise until it is correct.

24. Grading Policy

Grades are based on students' demonstration of the course content standards. Students will be given multiple opportunities both formative and summative to demonstrate this. Rubrics will be provided for all formative and summative projects and presentations to ensure fair and consistent grading. Clear learning objectives and grading criteria will be communicated at the beginning of the course to ensure all students understand expectations.

Students with IEPs may be given modified grades if indicated in their IEP. Teacher will work with Special Education staff to modify the curriculum as appropriate. Instead of extra credit, all students will be given the opportunity, with exception any final exam, to correct or improve their work for a better grade and improved understanding of the standards. If a student has *made a reasonable attempt* to complete an assignment the student will receive a grade no lower than 50% on Informal Assignments. The rational for this is to not be too punitive for low stakes assessments or assignments and to encourage students who are persisting through their struggle with the material.

Students' grades will be weighted mostly on summative assessment that is tied to the standards of the curriculum as follows:

70% Assessments

- End of unit Standards Based Tests & Quizzes
- Performance Tasks (Create & Explore)
- Summative Presentations
- Digital Portfolios
- Midterm and Final Exams

30% Checking for Understanding

- Small projects (formative)
- Reading & Homework
- Quizzes
- Warm ups
- Participation (class activities)