Mobile Computer Science Principles Syllabus

Overview:

The Mobile Computer Science Principles course provides an introduction to the basic principles of computer science (CS) from the perspective of mobile computing, including programming in App Inventor, a graphical programming language for Android mobile devices. The lessons and materials used by students incorporate programming while also integrating all other AP CSP big ideas: creativity, abstraction, data and information, algorithms, the internet and global impact. The curriculum engages students and supports the development of problem solving skills honing in on the computational thinking practices as indicated in the AP CSP curriculum framework. Students learn to create socially useful computational artifacts using App Inventor as well as connect computing and learn about abstracting as they develop and analyze their programs. The curriculum also emphasizes communication and collaboration in a project-based approach and classroom environment. This course involves a strong writing component. Students will maintain a portfolio of their work, which will include several performance tasks in the areas of programming and the impact of computing technology.

Prerequisites (As described by the College Board)

It is recommended that a student in the AP Computer Science Principles course should have successfully completed a first-year high school algebra course with a strong foundation in basic algebraic concepts dealing with function notation, such as $f(x) = 5\ x^2$ and problem-solving strategies that require multiple approaches and collaborative efforts. In addition, students should be able to use a Cartesian (x, y) coordinate system to represent points on a plane. It is important that students and their advisers understand that any significant computer science course builds upon a foundation of mathematical reasoning that should be acquired before attempting such a course.

Reference Text:

App Inventor 2: Create Your Own Android Apps. David Wolber, Hal Abelson, Ellen Spertus, and Liz Looney O'Reilly Media, Inc., 2014 (~$25 new on Amazon or view the Free Pre-publication Draft)

Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion. Hal Abelson, Ken Ledeen, Harry Lewis. Addison-Wesley, 2010 (Available via Free PDF Download)

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).

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*
- *Exam #2*

Assessments:

Portfolios

In this course students will document their work on their **portfolios**. That is, they will post answers to reading questions, write-ups of hands-on tutorials, written responses to assigned readings, and documentation of creative programming projects on their personal portfolio page. Each student will create a portfolio using Google sites (https://www.google.com/sites/overview.html). The portfolios will promote collaboration and sharing -- students can learn from each other -- and will constitute a full record of what the students have done in the course that they can refer back to during and after the course and share with their friends and family. Portfolios will be graded periodically 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. Brief, clear, and concise written responses to the study questions must be posted on students' portfolios.

Labs

This course will be taught in a computer lab. Students will have access to computers and mobile devices and any other necessary hardware, both during the class and during free periods. Students can work in the lab during their free periods. Internet access will be available to students throughout the course. In each unit, there will be at least three labs designed to practice and/or reinforce key concepts. Some are unplugged and others are completed in an online development environment. Most are completed in App Inventor.

Projects

There will be two (2) creative programming projects in which students will use lab time to work both individually and collaboratively (in pairs) to create a socially useful mobile app that they propose (pitch), design, and implement. One of these will be a practice for the College Board's Create Performance Task.

The second will be the official College Board Create Performance Task. Twelve (12) hours of class time will be provided for completion of the official Create Performance Task.

There will also be two (2) written research projects that students will work on individually. These research projects will focus on examining a computing innovation that has impacted society. One will be a practice for the Explore Performance Task. The second will be the College Board's Explore Performance Task. Eight (8) hours of class time will be provided for completion of the official College Board Explore Performance Task.

Oral and Video Presentations

There will be approximately three (3) oral and/or videotaped presentations of students' projects during the course.

Quizzes and Exams

There will be periodic quizzes, typically to wrap up the end of each unit, and a midterm exam given during the course. There will be a comprehensive final exam. Quizzes will be hand written and/or electronic and exams will be electronic.

Self-Check and Live Coding Exercises

All lessons in this course are accompanied by 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). These assessments are considered an essential part of the learning process. These are hosted online and may be done individually or with the class as a whole. Each question or exercise includes detailed feedback and students may repeat the question or exercise until it is correct.

AP CS Principles Exam

Students who complete this course will be prepared to take the AP CS Principles Exam.

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

Unit 1 of the course provides a brief overview of the Mobile CSP 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 environment, including their programming environment and online portfolios. Students are led through the process of creating a Gmail account, registering on the App Inventor site, and setting up their Google sites portfolio. Their portfolios will be used to display and share all of their written work for the course. Students are provided a brief introduction to blocks-based programming by having them work through a series of increasingly challenging Blockly Maze problems. And they are given a brief introduction to the *Blown to Bits* book, which is used as a reading resource throughout the course.

Guiding Questions:

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

Lessons: Welcome to Mobile CSP, **Mazes, Algorithms, and Programs,** Google Account and Portfolio Setup, App Inventor Setup, Blown to Bits (BB), Joining the Forum Wrap up	**Instructional Activity: Mazes, Algorithms, and Programs** The purpose of this activity is to show an example of what blocks-based programming is like and to introduce some basic terminology. Students are instructed to complete a sample ***Blockly*** activity in which they create small programs (*scripts),* using blocks, to solve mazes. The students are directed to the Angry Birds maze activity. After the teacher demonstrates the program, students may work alone or in pairs. This activity builds toward EU 4.1, EU 4.2, EU 5.1 and EU 5.2 by focusing on algorithm and programming concepts. LOs 4.1.1 [P2], 4.1.2 [P5], 4.2.4 [P4], 5.1.2 [P2], 5.1.3 [P6], 5.2.1 [P3]
Labs: Mazes, Algorithms, and Programs (Blockly), App Inventor Setup (App Inventor)	

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

Unit 2 provides an introduction to the App Inventor programming platform and the course's first programming project, the I Have a Dream app, a sound board app. Students are introduced to App Inventor's ***event-driven programming*** model. Students first work through a guided tutorial that plays an excerpt of a Martin Luther King speech and are then presented with several *exercises* that challenge them to extend their understanding by solving problems on their own, working in pairs. This is followed later in the unit by several *creative mini projects* where students are invited to express their own ideas by developing their own ***computational artifacts***. Students are also introduced to several important CS Principles themes and topics. Two lessons focus on ***hardware*** and ***software*** concepts. The big idea of ***abstraction*** is introduced. Students 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?

Lessons:	**Instructional Activity: I Have a Dream Projects**
I Have a Dream Tutorial, I Have a Dream Part 2, Mobile Apps and Mobile Devices, **I Have a Dream Projects,** What is Abstraction, Blown to Bits: The Digital Explosion, Binary Numbers, Where is North (A compass app), Hardware and Software Abstractions, Wrap up	The ***I Have a Dream Projects*** lesson is the third and culmination of a series of three related lessons: students are invited to express their own ideas and implement their own enhancements and extensions to the app we've been studying. In the first lesson students follow an instructor-led tutorial on how to build a basic sound board app (I Have a Dream). The instructor introduces basic App Inventor programming concepts, including the event-driven programming model that is used throughout the course. In the second lesson, students are given a set of small but increasingly challenging exercises and encouraged to work ***collaboratively*** to figure out the solutions on their own. In this culminating lesson, students design and implement enhancements and extensions to the app, including, possibly, creating an entirely new *sound board app* based on their own ideas and interests. These activities build toward EU 1.1, EU 1.2, EU 1.3, EU 5.1 and EU 5.4 by focusing on creativity, abstraction, and programming concepts. LOs 1.1.1 [P2], 1.2.1 [P2], 1.2.1 [P2], 1.2.3 [P2], 1.2.4 [P6], 1.3.1 [P2], 5.1.1 [P2], 5.4.1 [P4]
Labs: I Have a Dream Tutorial (App Inventor), I Have a Dream Part 1 (App Inventor), I Have a Dream Projects (App Inventor), Where is North (Compass App using 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 a number of 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 is presented in four parts each of which is followed by a set of creative project exercises and challenges. This unit also introduces two other apps: *Magic 8 Ball* app, which provides a first introduction to *lists,* and *Map Tour,* which demonstrates how to incorporate external data into a mobile app. 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 a number of 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. These lessons are complemented nicely by a *Blown to Bits* reading that focuses on digital documents, including how information can be hidden inside images and other digital documents.

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 manage the complexity of a programming?

Lessons:	**Instructional Activity: Representing Images**
Paint Pot 1 (A finger painting app), Paint Pot 1 Projects, **Representing Images,** Blown to Bits: Electronic Documents, Paint Pot 2 (An introduction to variables), Paint Pot 2 Projects, Error Detection, Magic 8-Ball Tutorial and Projects, Parity Error Detection, Map Tour Tutorial and Projects, Wrap up	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 can be used to represent all types of data, including numbers, images, characters, and machine language instructions. This activity builds toward EU 3.3 as 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. LOs 2.1.1 [P3], 2.1.2 [P5], 3.3.1 [P4]
Labs: Paint Pot 1 (App Inventor), Paint Pot 1 Projects (App Inventor), Paint Pot 2 (App Inventor), Paint Pot 2 Projects (App Inventor), Magic 8-Ball (Using 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 they have been given smaller challenges. This programming task is a practice for the official Create programming performance task that will be submitted to the College Board. Students are given 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 are taught how to brainstorm their ideas and develop wireframes with storyboards to express those ideas. Students are asked to give a 1-2 minute elevator pitch of their app idea and receive feedback from the instructor and their classmates. In class time is given to develop, test, and debug their app. The instructor answers any questions and provides feedback along the way. While working on their app, students are shown how to and asked to 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 are shown how to 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 build toward EU 1.1, EU 1.2, EU 2.2, EU 4.1, EU 5.1, EU 5.2, EU 5.3, EU 5.4, and EU 5.5 by focusing on creativity, abstraction, algorithm, and programming concepts.

LOs 1.1.1 [P2], 1.2.1 [P2], 1.2.2 [P2], 1.2.3 [P2], 1.2.4 [P6], 1.2.5 [P4], 2.2.1 [P2], 2.2.2 [P3], 4.1.1 [P2], 4.1.2 [P5], 5.1.1 [P2], 5.1.2 [P2], 5.1.3 [P6], 5.2.1 [P3], 5.3.1 [P3], 5.4.1 [P4], 5.5.1 [P1]

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 *Android Mash* 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.* The activities in Unit 4 build toward EU 2.3 as students learn that models use abstractions, such as a pseudo random number generator (PRNG), to represent real word situations, in this case, the flipping of a coin; EU 3.3 as students learn how PRNG algorithms are used to model *randomness* inside a computer, such as with the *Coin Flip* app; EU 7.1 as students extend the app model to represent different types of coins, including a biased coin and a three--sided coin. This is followed by an experimental lesson where an app that repeatedly "flips" a coin is used to assess the quality of App Inventor's PRNG; EU 7.3 as students learn how one's privacy is impacted by developing technology and computing innovations; and EU 7.4 as 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?

Lessons:	**Instructional Activity #1: Coin Flip Experiment**
Android Mash Tutorial, Android Mash Projects, Coin Flip Simulation, **Coin Flip Experiment**, Pseudo Random Number Generators (PRNGs), Coin Flip Simulation Projects, Real World Models, **Blown to Bits: Privacy,** Wrap up	In the prior lesson, students write the Coin Flip app, which simulates flipping a coin. In this lesson students work **collaboratively** to conduct an experiment using a mobile app that models a coin flip to test the hypothesis that App Inventor's PRNG is a good model of random behavior. Using the app, students are instructed to "flip" a coin 100s of times and asked to record the data in a table. When completed, the students calculate the percentage of heads and tails, which, in a good model should approach 50:50. Afterwards each group **communicates** their results to the class and the class spends time reflecting on what they learn from the experiment and how it could be refined. LOs 2.3.1 [P3], 2.3.2 [P3] **Instructional Activity #2: Blown to Bits: Privacy** After learning about the importance of animation, simulations, and modeling in the computing world, students will read a chapter or excerpt from *Blown to Bits* that focuses on privacy issues. Guided reading questions are provided for the students to answer independently. When the students are finished, the class spends time **communicating** their ideas and discussing how one's privacy is impacted by the developing technology and computing innovations, using *Blown to Bits* and their personal experiences as references. LOs 3.3.1 [P4], 7.1.1 [P4], 7.3.1 [P4], 7.4.1 [P1]

Labs: Android Mash (App Inventor), Android Mash Projects (App Inventor), Coin Flip (App Inventor), Coin Flip Projects (App Inventor)

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

Up until this point students have read Blown to Bits chapters and excerpts, as well as, read and discussed articles 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. This task is a practice for the official Explore performance task that will be submitted to the College Board. Students are given 4-5 hours of class time to complete this activity.

Activity: Impact of a computing innovation

This activity 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 are also asked to cite their sources and are instructed about *plagiarism.* The instructor assigns each group member a prompt taken from the official Explore Performance Task to answer about the innovation. Each group member answers the prompts in a single Google document that is shared among the group. The group then works together to edit the entire document discussing changes that need to be made. 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 the chosen innovation. Students are asked to share their artifact with their group members. After completing this activity, the students are asked to reflect on the experience and to brainstorm at least three computing innovations they might want to research for the official Explore Performance Task

This activity builds toward EU 1.1, EU 1.2, EU 3.1, EU 3.3, EU 7.1, EU 7.3, EU 7.4, and EU 7.5 by focusing on creativity, data, and global impact concepts.

LOs 1.1.1 [P2], 1.2.1 [P2], 1.2.2 [P2], 1.2.3 [P2], 1.2.5 [P4], 3.1.3 [P5], 3.3.1 [P4], 7.1.1 [P4], 7.3.1 [P4], 7.4.1 [P1], 7.5.1 [P1], 7.5.2 [P5]

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?

<table>
<tr>
<td>Lessons:

What is an algorithm?
Logo Part 1,
Logo Part 2,
Search Algorithms,
Sort Algorithms,
Analyzing Algorithms,
The Pong Game,
Limits of Algorithms,
Debugging Pong,
Blown to Bits: Web Searches,
Wrap up</td>
<td>Instructional Activity #1: Logo Part 2
Students are provided an app that implements a simple version of Logo, a programming language that lets them draw shapes by moving an Android dude around a canvas. In its initial version, students are given very impoverished procedures -- i.e., a move procedure that only moves the Android by 10 pixels and a turn procedure that only turns right by 90°. Students complete a series of drawing exercises that lead them to see limitations of the impoverished procedures -- i.e., it is very difficult to draw simple shapes and some shapes, such as a triangle, are impossible to draw. Students are then introduced to 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. In this way students are led to see the close interplay between algorithms and procedures.

LOs 2.2.1 [P2], 2.2.2 [P3], 2.2.3 [P3], 4.1.1 [P2], 5.3.1 [P3], 5.5.1 [P1]

Instructional Activity #2: Limits of Algorithms
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.

LOs 4.2.1 [P1], 4.2.2 [P1], 4.2.3 [P1]</td>
</tr>
<tr>
<td colspan="2">Labs: Logo 1 (App Inventor), Logo 2 (App Inventor), The Pong Game (App Inventor), Debugging Pong (App Inventor)</td>
</tr>
</table>

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 build toward EU 3.1, EU 3.2, EU 7.1, EU 7.2, and EU 7.5 by focusing 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?

Lessons:	**Instructional Activity: Using Fusion Tables to Visualize Big Data**
Presidents Quiz Tutorial, Presidents Quiz Projects, Blown to Bits: Who Owns the Bits?, Lists of Lists, Persistent Data, Sharing Data on the Web, Data Persistence Projects, Big Data, **Using Fusion Tables to Visualize Big Data,** A Mobile Fusion Table App Wrap up	A fusion table is a Google cloud application that helps you manage, process, and visualize data. After taking a tour of the fusion table gallery, to see the various ways they can be used to process and visualize data, students are taught the basics of creating a fusion table and importing data into it. It is pointed out that governments and other large organizations are increasingly making their data publicly available. The lesson, which may extend over multiple class periods, culminates with a ***collaborative activity*** in which students, working in pairs or groups, download a big data set from data.gov of their chosing using search techniques, upload it into a fusion table and then formulate some questions about the dataset that they use fusion table constructs to answer. LOs 3.1.1 [P4], 3.1.2 [P6], 3.1.3 [P5], 3.2.1 [P1], 3.2.2 [P3], 7.1.2 [P4], 7.2.1 [P1], 7.5.1 [P1], 7.5.2 [P5]
Labs: Presidents Quiz (App Inventor), Presidents Quiz Projects (App Inventor), Lists of Lists (App Inventor), Data Persistence Projects (App Inventor), Fusion Table App (App Inventor and Google Fusion Tables)	

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 Wifi, 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?

<table>
<tr>
<td>Lessons:

What is the Internet?,
No Texting While Busy Tutorial,
Cloud Computing and Ethics,
How the Internet Works,
My Directions Tutorial,
Cryptography Basics,
Cryptography: Securing the Internet
Blown to Bits: Cryptography,
Socially Aware App: Broadcast Hub,
Wrap up</td>
<td>Instructional Activity #1: How the Internet Works

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.

These activities builds toward EU 6.1 and EU 6.2 by focusing on concepts around the Internet, how the Internet works and the systems it is built on..

LOs 6.1.1 [P3], 6.2.1 [P5], 6.2.2 [P4]

Instructional Activity #2: Cryptography: Securing the Internet

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. This activity builds toward EU 6.3 by focusing on how cybersecurity is made possible through encryption.

LO 6.3.1 [P1]</td>
</tr>
<tr>
<td colspan="2">Labs: No Texting While Busy (App Inventor), My Directions (App Inventor and Google Maps APIs), Broadcast Hub (App Inventor)</td>
</tr>
</table>