Mobile Computer Science Principles
National Center for CS Education
1200 Kenwood Avenue | Duluth, MN 55811
www.mobile-csp.org | info@mobile-csp.org

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 MIT App Inventor, a graphical programming language for Android and iOS mobile devices. The lessons and materials used by students incorporate algorithms and programming while also integrating all other AP CSP big ideas: creative development, data, algorithms and programming, computing systems and networks, and impact of computing. 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 MIT App Inventor as well as connect computing and learning about algorithms and data as they develop and analyze their programs. The curriculum also emphasizes communication and collaboration in a project-based approach. This course involves a strong writing component. Students will maintain a portfolio of their work, which will include several projects 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:

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

Online Resources:

The ***complete curriculum*** is hosted online and free of charge: course.mobilecsp.org. The course uses many freely available resources that are 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 Computational Solution Design, Algorithms and Program Development, Abstraction in Program

Development, Code Analysis, Computing Innovations, and Responsible Computing with the five CS Principles Big Ideas of Creative Development (CRD), Data (DAT), Algorithms and Programming (AAP), Computing Systems and Networks (CSN), and Impact of Computing (IOC).

- *Unit 1 - Getting Started: Preview & Setup*
- *Unit 2 - Introduction to Mobile Apps & Pair Programming*
- *Unit 3 - Creating Graphics & Images Bit by Bit*
- *Unit 4 - Animation, Simulation, & Modeling*
- *AP Create: Programming Performance Task #1 (Practice)*
- *Exam #1*
- *Unit 5 - Algorithms & Procedural Abstraction*
- *Unit 6 - Communication Through The Internet*
- *Unit 7 - Using and Analyzing Data & Information*
- *AP Create: Programming Performance Task #2*
- *Unit 8 - AP CS Principles Assessment Prep*
- *Exam #2*
- *Unit 9 - Beyond the AP CSP Exam*

Assessments:

Portfolios

In this course, students will document their work in a **portfolio**. 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. 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 made available to students throughout the course. Some labs are "unplugged", some include Process Oriented Guided Inquiry Learning (POGIL) http://cspogil.org/Home activities, and others are completed in an online development environment. Most are completed in MIT 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 or 720 minutes of class time will be provided for completion of the official AP Create Performance Task after Unit 7.

There will also be a written research project that students will work on individually. The research project will focus on exploring and investigating a computing innovation that has impacted society. To prepare

students for this project and the Explore stimulus questions that will be part of the AP CSP Exam, students will complete at least three activities as part of the Impacts of CS lessons throughout the course. This includes an activity on understanding what a computing innovation is, an activity on determining the effects that computing innovations have on society, and an activity on how computing innovations analyze data. As part of these activities and the project, students will have the opportunity to explore at least three computing innovations including a citizen science app of their choosing.

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 also be a comprehensive second (final) exam. Quizzes will be hand written and/or electronic and exams will be electronic. Students may be asked to set up accounts on AP Classroom so that formative assessments may be issued and used to provide feedback at the end of each unit or topic.

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 questions 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 answered correctly.

AP CS Principles Exam

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

Unit 1: *Getting Started: Preview and Set up (CRD, AAP, & IOC)*

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 and its structure closely aligns to the online curriculum. 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, along with current events and news articles as a reading resource throughout the course to learn about the impact of CS on society.

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, ● Impacts of CS (Blown to Bits), ● Successful Learning in Mobile CSP ● 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 CRD-2 and EU AAP-2 by focusing on algorithm and programming concepts. **(CTP 2)** **LOs: CRD-2.B, AAP-2.A**
Labs: Mazes, Algorithms, and Programs (Blockly), App Inventor Setup (App Inventor)	

Unit 2: *Introduction to Mobile Apps and Pair Programming (CRD, AAP, CSN, & IOC)*

Unit 2 provides an introduction to the MIT App Inventor programming platform and the course's first programming project, the I Have a Dream app, which is a soundboard 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 enhancement projects* where students are invited to express their own ideas by developing their own ***mobile apps***. Students are also introduced to several important CS Principles themes and topics. Two lessons focus on ***hardware*** and ***software*** concepts. Students get their first look at ***binary numbers*** learning how to count in binary and how to view number systems such as binary 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 and Soundboard Projects**
● I Have a Dream Tutorial, ● The Internet and the Cloud, ● I Have a Dream Part 2, ● Mobile Apps and Mobile Devices: Hardware and Software, ● Algorithm Basics, ● **I Have a Dream and Soundboard Projects,** ● What is Abstraction, ● Binary Numbers, ● Hardware Abstractions: Logic Gates, ● Impacts of CS, ● Wrap up	The ***I Have a Dream and Soundboard 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 determine 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 *soundboard app* based on their own ideas and interests. These activities build toward EU CRD-1, CRD-2, and AAP-2 by focusing on collaborative creative development to design a program, algorithms, and programming concepts. **(CTP 1, CTP 6)** **LOs: CRD-1.A, CRD-1.B, CRD-2.A, CRD-2.B, CRD-2.C, CRD-2.D, and AAP-2.F**
Labs: I Have a Dream Tutorial (App Inventor), I Have a Dream Part 2 (App Inventor), I Have a Dream and Soundboard Projects (App Inventor)	

Unit 3: *Creating Graphics & Images Bit by Bit (CRD, DAT, AAP, & IOC)*

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 concepts of a *variables, lists* and *data abstraction*. The main app in this unit, The *Paint Pot* app, a computational model 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 three parts each of which is followed by a set of creative project exercises and challenges. This unit also introduces two other apps: *Map Tour* app, which provides a first introduction to *lists,* and *Map Tour with TinyDB,* which demonstrates how to incorporate external data and location into a mobile app. Unit 3 also extends the student's understanding of the *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 by an *Impact of CS* reading and activity.

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 program?

Lessons:	**Instructional Activity: Representing Images**
• Paint Pot Tutorial (A finger painting app with variables), • **Representing Images,** • Paint Pot Projects, • Paint Pot Refactoring and Procedural Abstraction, • Error Detection, • Parity Error Checking, • Map Tour Tutorial, • Map Tour with TinyDB • Impacts of CS, • 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, colors, and machine language instructions. This activity builds toward DAT-1 as students learn about *lossy* and *lossless* compression algorithms and 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: DAT-1.A and DAT-1.D**
Labs: Paint Pot Tutorial (App Inventor), Paint Pot Projects (App Inventor), Paint Pot Refactoring and Procedural Abstraction (App Inventor), Map Tour (App Inventor), Map Tour with TinyDB (App Inventor and Google Maps)	

Unit 4: *Animation, Simulation, and Modeling (CRD, DAT, AAP, & IOC)*

Unit 4 focuses on *animation, simulation and modeling*. The *LightsOff* app introduces the idea of *computer simulation* with a computational variation 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 AAP-3 as students learn that models and simulations 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. This is followed by an experiment lesson where an app that repeatedly "flips" a coin is used to assess the quality of App Inventor's PRNG; Students learn how one's privacy is impacted by developing technology and computing innovations; and 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: Impacts of CS**
● LightsOff Tutorial, ● LightsOff Projects, ● Logo Part 1, ● Coin Flip Simulation Tutorial, ● Coin Flip Experiment, ● Pseudo Random Numbers, ● Real World Models, ● Abstraction: Inside the CPU, ● **Impacts of CS,** ● Wrap up	After learning about the importance of animation, simulations, and modeling in the computing world, students will learn about the *Impact of CS* by reading an article, chapter, or current event 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 technology and computing innovations, using the reading and their personal experiences as references. In this lesson, students also complete the "Analyzing Data in Computing Innovations" activity from the College Board's Explore Curricular Requirement Resources and examine how privacy issues have become a concern with computing innovations such as electronic health records, thus building toward EU IOC-2. **(CTP 5)** **LOs: IOC-2.A and IOC-2.B**
Labs: LightsOff Tutorial (App Inventor), LightsOff Projects (App Inventor), Logo Part 1 (App Inventor), Coin Flip Simulation Tutorial (App Inventor)	

Create: Programming Performance Task #1 *(CRD & AAP)*

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 6 hours of class time to complete this task.

Practice 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 development process including any progress made and any challenges or opportunities they may have faced. Students describe major functionality of their apps using screenshots of blocks of code, along 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.

Unit 5: Algorithms and Procedural Abstraction (*AAP & IOC)*

In Unit 5, algorithms and procedures are examined in more detail. The Logo apps, started in Unit 4 and continued in Unit 5, introduce the concept of procedural abstraction and students learn to define and use procedures -- named blocks of code that perform a specific task -- as well as when and how to use procedures with parameters. 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.*** Students compare various searching and sorting algorithms, including examining the efficiency of each. For the searching and sorting algorithms, students analyze them both experimentally and through mathematical concepts such as functions and graphs **(CTP 4)**. The impact section of this unit provides another opportunity for students to explore a computing innovation, such as web searching, and the effects it has on society.

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?

Lessons: ● **Logo Part 2,** ● Search Algorithms, ● Sorting Algorithms, ● Quiz App, ● Quiz App Projects, ● Analyzing Algorithms, ● **Limits of Algorithms,** ● Parallel Computing ● Impacts of CS, ● Wrap up	**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 icon 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. **(CTP 3)** **LOs: AAP-2.L, AAP-3.A, AAP-3.B, and AAP-3.C** **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: AAP-4.A and AAP-4.B**
Labs: Logo Part 2 (App Inventor), Quiz App (App Inventor), Quiz App Projects (App Inventor)	

Unit 6: *Communication Through The Internet (CRD, AAP, CSN, & IOC)*

Unit 6 focuses on ***Computing Systems and Networks***, one of the big ideas in computer science. The App Inventor lesson in this unit shows how to use the Internet in apps, including the ability to send text messages over Wifi. The CS Principles lessons focus on the Internet, how it works, how it enables

innovation and collaboration, and security concerns for using it. In this unit, students complete a series of activities using network administration software tools such as *Ping* and *Traceroute* as well as use a Domain Name System (DNS) simulator app to explore how we communicate on the Internet with IP addresses.

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:

● Computer Networking,
● Network Architecture,
● IP Addresses and Domain Names,
● Caesar Cipher App,
● Cryptography Basics,
● Cryptography: Securing the Internet
● Debugging Caesar Cipher,
● Computer Security
● Impacts of CS,
● Wrap up</td>
<td>Instructional Activity #1: Network Architecture

This lesson goes more deeply into the infrastructure and mechanics of computer networks. 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.

These activities builds toward EU CSN-1 by focusing on concepts around the Internet, how the Internet works and the hardware, algorithms, and protocol systems it is built on.

LOs: CSN-1.A, CSN-1.B, CSN-1.C, and CSN-1.E

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 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 IOC-2 by focusing on how cybersecurity is made possible through encryption.

LOs: IOC-2.B</td>
</tr>
<tr>
<td colspan="2">Labs: Caesar Cipher App (App Inventor), Debugging Caesar Cipher (App Inventor)</td>
</tr>
</table>

Unit 7: *Using and Analyzing Data and Information (CRD, DAT, AAP, & IOC)*

Unit 7 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 databases, such as Google Firebase. This unit's CS Principles lessons build toward EU DAT-2 and EU IOC-1 by focusing on the concept of Big Data and its growing importance. Students are also introduced to some of the algorithms needed to process massive datasets efficiently.

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: Visualizing Data**
● Big Data, ● **Visualizing Data,** ● Data Visualization Project, ● Data Map App ● Clicker App with CloudDB ● Artificial Intelligence and Machine Learning, ● Impacts of CS, ● Wrap up	A Google spreadsheet is a cloud application that helps you manage, process, and visualize data. After taking a tour of Google spreadsheets, to see the various ways they can be used to process and visualize data, students are taught the basics of using data visualization tools. It is pointed out that governments and other large organizations collect a lot of data on individuals. The lesson, which may extend over multiple class periods, culminates with a ***collaborative activity*** in which students, working in pairs or groups, analyze a data set. The data set is chosen using search techniques, and uploaded into a Google spreadsheet. The students formulate some questions about the dataset and then use visualization tools to answer them. **LOs: DAT-2.A, DAT-2.B, DAT-2.D, and DAT-2.E**
Labs: Visualizing Data (Google Spreadsheets), Data Visualization Project (Google Spreadsheets) Data Map App (App Inventor), Clicker App with CloudDB (App Inventor)	

Unit 8 - AP CS Principles Assessment Prep

- About the AP CS Principles Exam
- AP CSP Vocabulary Review
- AP CSP Pseudocode
- Tracing Pseudocode Exercises
- Sample AP CSP Exam Questions
- The Mobile CS Principles Quiz App
- Additional Resources
- Create PT Samples

Unit 9 - Beyond the AP CSP Exam

- Careers in IT and CS
- Films in Computer Science
- Transitioning to Text-Based Languages and CS A
- Magic 8-Ball Tutorial and Projects (App Inventor)
- Persisting Photos Tutorial and Projects (App Inventor)
- Where is North: A Compass App (App Inventor),
- My Directions Tutorial (App Inventor and Google Maps APIs)
- The Pong Game (App Inventor)
- Debugging Pong (App Inventor)
- Multiple Choice Quiz App: List of Lists (App Inventor)
- No Texting While Busy Tutorial (App Inventor)
- Socially Aware App: Broadcast Hub Tutorial (App Inventor)
- Map Tour Tutorial with Activity Starter and Google Maps (App Inventor)

Sample Course Schedule for a Full Year Course

Unit	Estimated Start Date	Estimated Timing
Unit 1	End of August	180 Minutes - Four 45 min class periods
Unit 2	Beginning of September	585 Minutes - Thirteen 45 min class periods
Unit 3	Beginning of October	585 Minutes - Thirteen 45 min class periods
Unit 4	Beginning of November	585 Minutes - Thirteen 45 min class periods
Create #1*	Mid December	360 Minutes - Eight 45 min class periods
Exam 1 - Midterm*	Mid January	135 Minutes - Two 45 min reviews, One 45 min exam
Unit 5	Mid January	585 Minutes - Thirteen 45 min class periods
Unit 6	End of February	675 Minutes - Fifteen 45 min class periods
Unit 7	Mid March	495 Minutes - Eleven 45 min class periods
Create #2	Beginning of April	720 Minutes - Sixteen 45 min class periods
Unit 8	Beginning of May	135 Minutes - Three 45 min class reviews
Exam 2 - Final	Beginning of May	45 Minutes - One 45 min exam
AP CSP Exam	Beginning of May	Two hour long exam
Unit 9	Mid May	Extra App Inventor lessons as well as suggestions for other resources to engage students with future CS courses, majors, and careers

*Note: The order of these two units/assessments can be switched around to fit your particular exam and winter break schedule.