CodeHS

Georgia Computer Science Principles in JavaScript
Course Syllabus

Introduction

Georgia Computer Science Principles introduces students to the foundational concepts of computer science and programming in JavaScript. With a unique focus on creative problem solving and real-world applications, students are challenged to explore how computing and technology can impact the world.

Course Overview

Prerequisites: There are no official prerequisites for the Georgia Computer Science Principles course. This course is meant to be a first-time introduction to computer science and does not require students to come in with any computer programming experience.

Learning Environment: The course utilizes a blended classroom approach. The content is a mix of web-based and physical activities. Students will write and run code in the browser, create websites and digital artifacts, and engage in in-person collaborative exercises with classmates. Teachers utilize tools and resources provided by CodeHS to leverage time in the classroom and give focused 1-on-1 attention to students. Each unit of the course is broken down into lessons. Lessons consist of video tutorials, short quizzes, example programs to explore, written programming exercises, free response exercises, collaborative creation projects, and research projects.

Programming Environment: Students write and run programs in the browser using the CodeHS editor. Students will be able to write both text-based and block-based JavaScript programs. Students gain programming experience early on in the course that will enable them to explore the rest of the course topics through computational thinking practices.

Course Resources: Access to a computer and high-speed internet is required. There is also an online textbook available for many modules and topics which can be accessed through the lesson plans or at https://codehs.gitbooks.io/introcs/content/

Course Breakdown

Unit 1: Introduction to Programming with Karel the Dog (3 weeks, 15 hours)
This course begins with a strong focus on programming in order to allow students to create computational artifacts early on in the course. Students will be able to use their knowledge of programming to explore future topics in the course.

We use Karel, a dog that only knows how to move, turn left, and place tennis balls in his world, to show students what it means to program, and allow students to focus on computational problem-solving. Students will learn about the need for programming languages, the uses of programs, how to write programs to solve computational problems, how to design algorithms, how to analyze and compare potential solutions to programming problems, and learn the value and challenges involved in collaborating with others to solve programming problems. Students will use the grid coloring functionality of Karel to create a digital painting and embed this program in their portfolio website.

Subsection	Lessons / Topics
Abstraction	Procedural Abstraction Modularity Program Reuse Digital Data (Bits) Reducing Complexity
Programming Style	Program Documentation Using Existing Code and Libraries APIs Commenting Code
Control Structures	If/Else Statements (Selection) For Loops and While Loops (Iteration)
Debugging Strategies	Logic Errors Syntax Errors Run-Time Error Testing
Designing Algorithms	Sequencing, Selection, Iteration Clarity and Readability Using Existing Algorithms

	Optimization and Efficiency
Example Activities and Big Idea/Computational Thinking Practice *The Two Towers:* In this program, students have Karel build two towers of tennis balls. Each tower should be 3 tennis balls high. In the end, Karel should end up on top of the second tower, facing East. Students need to write at least 3 functions in order to solve this problem. This activity requires students to design and create functions for repeated processes within their program. Students need to consider top-down design and decomposition through the following questions: ● How can you break this problem down into smaller problems? ● What is a subtask that Karel needs to do more than once in this problem? **[Big Idea AAP][Computational Thinking Practice 1]**	

Unit 2: Practice PT: Pair-Programming Paint (3 days, 3 hours)
Students will use the grid coloring functionality of Karel to create a digital image. They will then embed this Karel program into their personal website portfolio.

Subsection	Lessons / Topics
Collaboration and Communication	Collaboration Diverse Perspectives Bias Avoidance Pair-Programming Design and Planning Program Documentation Acknowledgement of Reused Code
Example Activity and Big Idea/Computational Thinking Practice *Create Your Own UltraKarel Image:* Following the milestones and the pseudocode plan that students have laid out, students use pair-programming to write the code for their final project. They then test their code along the way to make sure they have solved each milestone. This activity allows students to develop something completely unique with their programming skills and implement a successful algorithm of their own design. Students then reflect upon and answer the following questions:	

1. Identify the programming language and purpose of your program.

2. Describe the incremental and iterative development process of your program. How did you divide the program into smaller tasks and make a plan to complete them all?

3. Describe the difficulties and/or opportunities you encountered and how they were resolved or incorporated.

4. Identify an algorithm that is fundamental for your program to achieve its intended purpose and includes two or more additional algorithms.

5. Describe how each algorithm within your selected algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program.

6. Identify an abstraction you developed, and explain how your abstraction helped manage the complexity of your program.
[Big Idea CRD][Computational Thinking Practice 2]

Unit 3: Programming with JavaScript (2 weeks, 10 hours)
This unit introduces students to the basics of JavaScript, including variables, user input, control structures, functions with parameters and return values, and basic graphics, how to send messages to objects.

Subsection	Lessons / Topics
Programming Languages Lessons: *What is Code?* *Uses of Programs*	What is Programming? Pseudocode Programming Languages Computing Innovations
Variables Lessons:	Variable Names Assignment Operators Data Types Variables as Abstractions

Variables	
Arithmetic Expressions Lessons: *Basic Math in JavaScript*	Program Behavior Testing using Inputs Arithmetic Expressions Order of Operations Modulus String Concatenation
User Input Lessons: *User Input* *Mouse Events: Mouse Clicked* *Key Events*	Strings User Input Program Output Events Mouse and Key Events

Example Activity and Big Idea/Computational Thinking Practice
Computing Innovations (as part of *Uses of Programs* lesson): In this activity, students perform an online search for examples of computing innovations that have had an impact on society, economy, or culture. The computing innovations must consume, produce, and/or transform data. A computing innovation can be a physical object like a self-driving car, non-physical software like a picture editing software, or a non-physical concept like e-commerce.

Students
- practice searching and evaluating sources relevant to computing innovations
- write the definition of *computing innovation* in their own words
- list 5 items that ARE computing innovations and 5 items that are NOT computing innovations. For each one, explain the reason why it is or is not a computing innovation
- identify the data used in at least one computing innovation and explain how the data is consumed, produced, or transformed by the given computing innovation. **[Computing Innovation 1, Prompt B][Big Idea IOC][Computational Thinking Practice 5]**

Unit 4: JavaScript Control Structures (2 weeks, 10 hours)
In this unit, students learn how to use booleans and logical operators with control structures to make more advanced programs in JavaScript.

Subsection	EKs	Lessons / Topics
Comparison Operators <u>Lessons</u>: *Booleans* *Comparison Operators*	AAP-2.E.1 AAP-2.F.4 AAP-2.E.2 AAP-2.F.5 AAP-2.F.1 AAP-2.F.2 AAP-2.F.3	Booleans Relational Operators Operands
Selection <u>Lessons</u>: *If Statements* *Random Numbers*	AAP-2.G.1 AAP-2.I.2 AAP-2.H.1 AAP-2.L.3 AAP-2.H.2 AAP-2.L.4 AAP-2.H.3 AAP-3.E.2 AAP-2.I.1	Selection Conditional Statements Nested Conditionals Equivalent Boolean Statements Random Numbers
Iteration <u>Lessons</u>: *While Loops*	AAP-2.K.2 AAP-2.L.1 AAP-2.K.3 AAP-2.L.2 AAP-2.K.4 AAP-2.L.5 AAP-2.K.5	Iteration Loops Different but Equivalent Algorithms
Example Activity and Big Idea/Computational Thinking Practice *Better Password Prompt:* Students write a program that uses a while loop to prompt a user for a password. They keep prompting the user for the password, and if they get it correct, they then break out of the loop. If they don't get it correct, they should give the user an error message. This activity requires that students use multiple program statements in a specific order to solve a problem. **[Big Idea AAP][Computational Thinking Practice 2]**		

Unit 5: Functions and Parameters (2 weeks, 10 hours)
In this unit, students learn to write reusable code with functions and parameters.

Subsection	EKs	Lessons / Topics
Functions and Parameters <u>Lessons</u>: *Functions and Parameters 1* *Functions and Parameters 2* *Functions and Return Values 1* *Functions and Return Values 2*	CRD-2.C.6 AAP-3.A.3 CRD-2.D.2 AAP-3.A.4 CRD-2.B.3 AAP-3.B.5 CRD-2.C.4 AAP-3.C.1 AAP-3.A.1 AAP-3.C.2 AAP-3.A.2 AAP-2.M.2	User and Application Input Program Output Procedures Parameters Return Values Using Existing Algorithms

Example Activity and Big Idea/Computational Thinking Practice
Pool Table: Students write a program with a function that draws a pool ball. This function should take as parameters, the color, the number that should go on the pool ball, and the location of the center of the pool ball. Students need to consider the function abstractly as a means for taking specific data via the parameters and creating a unique graphical output based on those inputs.
[Big Idea DAT][Computational Thinking Practice 3]

Unit 6: Practice PT: Tell a Story (3 days, 3 hours)
In this project, students will write a JavaScript program that tells a graphical story

Example Activity and Big Idea/Computational Thinking Practice
Tell a Story! In this activity, students write a JavaScript program that tells a graphical story in at least 4 scenes. Following the milestones and the pseudocode plan that students have laid out prior to this exercise, students write the code for their final project. They iterate and test their code along the way to make sure they have solved each milestone.
[Big Idea CRD][Computational Thinking Practice 4]

Unit 7: Basic Data Structures (2 weeks, 10 hours)
In this unit, students learn to write reusable code with functions and parameters.

Subsection	EKs	Lessons / Topics
Basic Data Structures Lessons: *Intro to Lists/Arrays* *Indexing Into an Array* *Removing an Element*	DAT-1.A.1 AAP-1.A.1 AAP-1.C.1 AAP-1.C.2 AAP-1.C.3 AAP-1.D.6 AAP-1.D.7 AAP-1.D.8 AAP-2.N.2 AAP-2.N.1	Data Values Lists and Elements Indices List Procedures
Data Abstractions Lessons: *Adding/Removing From Arrays* *Array Length and Looping*	AAP-1.D.1 AAP-1.D.5 DAT-2.E.4 AAP-1.D.2 AAP-1.D.3 AAP-1.D.4 DAT-2.E.2	Data Abstraction Translating and Transforming Data Filtering and Cleaning Patterns

	DAT-2.D.4 DAT-2.E.5	
Traversing a List <u>Lessons</u>: *Array Length and Looping* *Iterating Over an Array* *Removing an Element*	DAT-2.D.6 AAP-2.O.1 AAP-2.O.2 AAP-3.C.1 AAP-3.C.2 AAP-3.A.6 AAP-2.O.3 AAP-3.A.5 AAP-3.A.7 AAP-3.A.8 AAP-3.E.1	Extract and Modify Information Traversing a List Iteration Statements
Algorithm Efficiency <u>Lessons</u>: *Array Length and Looping* *Finding an Element in a List*	AAP-2.O.4 DAT-2.D.3 AAP-2.O.5 AAP-2.P.1 AAP-2.P.2 AAP-2.P.3 AAP-4.A.1 AAP-4.A.3 AAP-4.A.7 AAP-4.A.8 AAP-4.A.9	Using Existing Algorithms Search Tools Linear Search Binary Search Algorithm Efficiency Heuristics
SImulation <u>Lessons</u>: *Simulation*	AAP-3.F.1 AAP-3.F.2 AAP-3.F.3 AAP-3.F.4 AAP-3.F.5 AAP-3.F.6 AAP-3.F.7 AAP-3.F.8	Simulations as Abstractions Bias in Simulations Random Number Generators

Example Activity and Big Idea/Computational Thinking Practice
Draw a Barcode: Students write a program to draw a barcode on the screen given an array that represents the data in the barcode. The array will contain a boolean in it, and if the boolean is `true`, the program will need to draw a vertical line in that position that runs from the top to the bottom of the screen. If not, the program will not draw a line. This program development requires students to use data generated from their bit array and loops and conditionals to determine where lines are drawn and where they are not drawn.
[Big Idea DAT][Computational Thinking Practice 2]

Unit 8: Digital Information (3 weeks, 15 hours)

In this unit, students will learn about the various ways we represent information digitally. Topics covered include number systems, encoding data, programmatically creating pixel images, comparing data encodings, compressing and encrypting data. Students will work in pairs to develop their own data encryption algorithms and attempt to crack the encryptions of their peers. Their text encryption tool will be embedded in their portfolio websites.

Subsection	EKs	Lessons / Topics
Number Systems Lessons: *Intro to Digital Information* *Number Systems*	CRD-2.C.1 DAT-1.A.7 CRD-2.D.1 DAT-1.B.1 CRD-2.J.2 DAT-1.B.2 CRD-2.J.3 DAT-1.B.3 CRD-2.I.4 DAT-1.C.1 DAT-1.A.2 DAT-1.C.2 DAT-1.A.3 DAT-1.C.3 DAT-1.A.4 DAT-1.C.4 DAT-1.A.5 DAT-1.C.5 DAT-1.A.6	Computing Devices Abstraction Program Input and Output Bits and Bytes Overflow Errors Range of Value Limits Binary and Decimal Systems
Data Compression Lessons: *Data Compression* *Lossy Compression*	DAT-1.A.8 DAT-1.D.4 DAT-1.A.9 DAT-1.D.5 DAT-1.A.10 DAT-1.D.6 DAT-1.D.1 DAT-1.D.7 DAT-1.D.2 DAT-1.D.8 DAT-1.D.3	Lossless Data Lossy Data Digital and Analog Data
Cryptography Lessons: *Cryptography*	AAP-4.B.1 AAP-4.B.2 AAP-4.B.3 IOC-2.B.8 IOC-2.B.5	Decidable Problems Computer Viruses Encryption

Example Activity and Big Idea/Computational Thinking Practice
Guess the Passcode: Students first imagine they forgot their 4-digit passcode for their phone, and need to guess the correct passcode. They develop a program to guess passcodes for them to speed up the process. Once the correct passcode has been guessed, the program should print out how many guesses it took to reach the correct one. This activity encourages students to consider security issues which can be expanded to how we create a safer computing culture.

Students discuss the following questions with a partner:
1. How many possible passcodes will you need to guess before you've guessed every possible passcode?
2. Why is this dangerous for the security of your phone?

3. Imagine a hacker had access to your phone and had written a program to guess every possible passcode until they had broken in. What defenses could we build into the phone to keep this guess and check strategy from working? (What happens when you guess incorrectly over and over again?)
4. Can you think of any guessing strategies that might be faster than starting at 0000 and iterating all the way up to 9999

[Big Idea IOC][Computational Thinking Practice 6]

Unit 9: Practice PT: Steganography (3 days, 3 hours)
In this project, students will be implementing a form of cryptography known as Steganography. Students can choose this practice PT or the following.

Example Activity and Big Idea/Computational Thinking Practice
Secret Image: Steganography- Students use a form of cryptography called steganography to hide a secret image inside of a cover image. They need to develop two functions that create filters, with one encoding and the other decoding. They are required to use a solid degree of abstraction since several functions will be required for each part of the encoding and decoding process. This also continues their consideration and discussions of privacy issues in computing.
[Big Idea IOC][Computational Thinking Practice 3]

Unit 10: Practice PT: Create Your Own Image Filter (3 days, 3 hours)
In this project, students pair up with a partner to develop a novel image filter that can be applied to any digital image of their choosing. They will describe their image filter, and their development process, and embed their image filter along with its description on their personal portfolio website. Students can choose this practice PT or the previous.

Example Activity and Big Idea/Computational Thinking Practice
Create an Image Filter: In this activity, students work with a partner to develop functions for creating unique mage filters. They share their creative solutions designs with others and incorporate feedback for improvement.
[Big Idea CRD][Computational Thinking Practice 1]

Unit 11: The Internet (2 weeks, 10 hours)
This unit explores the structure and design of the internet, and how this design affects the reliability of network communication, the security of data, and personal privacy. Students will learn about the protocols and algorithms used on the internet and the importance of cybersecurity. Students will choose an innovation that was enabled by the Internet and explore the positive and negative impacts of their innovation on society, economy, and culture. Students

will develop a computational artifact that illustrates, represents, or explains the innovation's purpose, its function, or its effect, and embed this artifact in their personal portfolio website.

Subsection	EKs	Lessons / Topics
Internet Hardware and Addresses <u>Lessons</u>: *Welcome to the Internet* *Internet Hardware* *Internet Addresses*	CSN-1.A.1 CSN-1.A.8 CSN-1.A.2 CSN-1.B.3 CSN-1.A.3 CSN-1.B.4 CSN-1.A.4 CSN-1.A.7	Protocols Computing Devices Computer Networks Bandwidth
Routing <u>Lessons</u>: *Routing*	CSN-1.A.5 CSN-1.E.2 CSN-1.A.6 CSN-1.E.3 CSN-1.B.5 CSN-1.E.4 CSN-1.B.6 CSN-1.E.5 CSN-1.B.7 CSN-1.E.6 CSN-1.E.1 CSN-1.E.7	Routing Scalability Fault-Tolerance Redundancy
Packets and Protocols <u>Lessons</u>: *Packets and Protocols*	CSN-1.B.1 CSN-1.D.1 CSN-1.B.2 CSN-1.D.2 CSN-1.C.1 CSN-1.D.3 CSN-1.C.2 DAT-2.B.1 CSN-1.C.3 DAT-2.B.3 CSN-1.C.4 DAT-2.B.5	Datastreams Packets IP, TCP, UDP HTTP Metadata
Computing Systems <u>Lessons</u>: *Sequential, Parallel & Distributed*	DAT-2.C.7 CSN-2.A.6 DAT-2.C.8 CSN-2.A.7 CSN-2.A.1 CSN-2.B.1 CSN-2.A.2 CSN-2.B.2 CSN-2.A.3 CSN-2.B.3 CSN-2.A.4 CSN-2.B.4 CSN-2.A.5 CSN-2.B.5	Parallel Systems Scalability of Systems Sequential Computing Parallel Computing Distributed Computing Efficiency of Solutions Speedup
Impact of the Internet <u>Lessons</u>: *The Impact of the Internet* *Creative Credit and Copyright*	IOC-1.A.1 IOC-1.E.2 IOC-1.A.3 IOC-1.E.3 IOC-1.A.4 IOC-1.E.4 IOC-1.A.5 IOC-1.E.5 IOC-1.B.1 IOC-1.E.6 IOC-1.B.2 IOC-1.F.1 IOC-1.B.3 IOC-1.F.2 IOC-1.B.4 IOC-1.F.3 IOC-1.B.5 IOC-1.F.4 IOC-1.B.6 IOC-1.F.5	Computing Innovations Unintended Effects Impact on Society Rapid Sharing Digital Divide Citizen Science Crowdsourcing Creative Credit and Copyright

	IOC-1.C.1 IOC-1.F.6 IOC-1.C.2 IOC-1.F.7 IOC-1.C.3 IOC-1.F.9 IOC-1.C.4 IOC-1.F.10 IOC-1.C.5 IOC-1.F.11 IOC-1.E.1	
Cybersecurity <u>Lessons</u>: *Cybersecurity*	IOC-1.F.8 IOC-2.B.5 IOC-2.A.1 IOC-2.B.6 IOC-2.A.7 IOC-2.B.7 IOC-2.A.8 IOC-2.B.9 IOC-2.A.9 IOC-2.B.10 IOC-2.A.11 IOC-2.B.11 IOC-2.A.12 IOC-2.C.1 IOC-2.A.13 IOC-2.C.2 IOC-2.A.15 IOC-2.C.3 IOC-2.B.1 IOC-2.C.4 IOC-2.B.2 IOC-2.C.5 IOC-2.B.3 IOC-2.C.6 IOC-2.B.4 IOC-2.C.7	Legal and Ethical Concerns Personally Identifiable Info (PII) Digital Footprint Authentication Certificate Authorities (CAs) Computer Viruses Malware Phishing Keylogging Rogue Access Points Encryption

Example Activity and Big Idea/Computational Thinking Practice
Reflection: Unintended Effects - Students consider the WWW, targeted advertising and machine learning and data mining as examples of computing innovations. They also learn that responsible programmers try to consider the unintended ways their computing innovations can be used and the potential beneficial and harmful effects of these new uses although it may not be possible for a programmer to consider all the ways a computing innovation can be used.

They then consider *Pokemon Go* (from the previous video) or research another innovation that had unintended effects. Students answer in their reflections:

1. What were the intended effects and what were the unintended effects?
2. Explain beneficial and harmful effects of at least one other computing innovation on society, economy, or culture.

[Computing Innovation 2, Prompt A][Big Idea IOC][Computational Thinking Practice 5]

Packets and Protocols: The Story of the Internet - In their own words, students tell the story of downloading an image from a website on the internet. They tell the story step by step of how their computer finds the relevant server, requests information from the server, and receives it. Students are required to include distinctions between the internet and the World Wide Web, such as:
- The World Wide Web is a system of linked pages, programs, and files.
- HTTP is a protocol used by the World Wide Web.

- The World Wide Web uses the Internet.

[**Big Idea CSN**][**Computational Thinking Practice 5**]

Unit 12: Practice PT: The Effects of the Internet (3 days, 3 hours)
In this project, students will choose an innovation that was enabled by the Internet and explore the positive and negative impacts of their innovation on society, economy, and culture. Students will develop a computational artifact that illustrates, represents, or explains the innovation's purpose, its function, or its effect, and embed this artifact in their personal portfolio website.

Example Activity and Big Idea/Computational Thinking Practice
The Effects of the Internet: Students provide evidence of the extensive knowledge they have developed about a chosen Internet-based innovation and its impact(s). Students include citations, as applicable, within their written responses.

Within their computational artifact, students explain at least one beneficial effect and at least one harmful effect the Internet-based innovation has had, or has the potential to have, on society, economy, or culture. They also identify data privacy, security, or storage concerns for the computing innovation.
[**Computing Innovation 3, Prompt C**][**Big Idea IOC**][**Computational Thinking Practice 5**]

Unit 13: Data (1 week, 5 hours)
In this unit, students will explore using computational tools to store massive amounts of data, manipulate and visualize data, find patterns in data, and draw conclusions from data. Students will consider how the modern wealth of data collection has impacted society in positive and negative ways. Students will work in teams to investigate a question of personal interest and use public data to present a data-driven insight to their peers. They will develop visualizations to communicate their findings, and embed their visualizations in their portfolio websites.

Subsection	EKs	Lessons / Topics
Visualizing and Interpreting Data Lessons: *Getting Started with Data* *Visualizing and Interpreting Data*	DAT-2.A.1 DAT-2.D.5 DAT-2.A.2 DAT-2.D.6 DAT-2.C.1 DAT-2.E.1 DAT-2.D.1 DAT-2.E.2 DAT-2.D.2 DAT-2.E.3 DAT-2.D.3 DAT-2.E.5 DAT-2.D.4	Filtering and Cleaning Data Patterns and Trends Search Tools Tables, Diagrams and Displays Interactive Visualizations Combining Data Sources
Collecting Data and Data Limitations	DAT-2.A.3 DAT-2.C.2 DAT-2.A.4 DAT-2.C.3	Metadata Correlation

Lessons: *Data Collection and Limitations*	DAT-2.B.1 DAT-2.C.4 DAT-2.B.2 DAT-2.C.5 DAT-2.B.3 DAT-2.C.6 DAT-2.B.4 DAT-2.D.6 DAT-2.B.5 CRD-2.F.3	Using a Variety of Sources Incomplete or Invalid Data Bias Surveys, Testing, Interviews

Example Activity and Big Idea/Computational Thinking Practice
Importance of Metadata: Students consider how metadata can increase the effective use of data or data sets by providing additional information. They consider the importance of metadata and reflect on why metadata is important for a data set, how metadata help in finding specific data, and what metadata should reveal about the data.
[Big Idea DAT][Computational Thinking Practice 5]

Unit 14: Practice PT: Present a Data-Driven Insight (3 days, 3 hours)
In this project, students will work with a partner to answer a question of personal interest using a publicly available data set. Students will need to produce data visualizations and explain how these visualizations led to their conclusions. They will develop a computational artifact that illustrates, represents, or explains their findings, communicate their findings to their classmates, and embed their artifact in their personal portfolio website.

Example Activity and Big Idea/Computational Thinking Practice
Present a Data-driven Insight: Students consider how the amount of collected data impacts our lives in ways that require considerable study and reflection for us to fully understand them. Students explore a question that can be answered by analyzing a dataset. They form a question and use visualization techniques to analyze the data to answer the question.
[Big Idea DAT][Computational Thinking Practice 6]

Unit 15 & 16: Explore MCQ Practice and Create Performance Task (3 weeks, 15 hours)
This time is set aside for students to prepare for the Explore MCQ and create their AP Create Performance Task. Students will be given the chance to review course content and practice the skills necessary to complete the Create Performance Task. The Create PT will be administered over 12 hours of class time.

Subsection	EKs	Lessons / Topics
AP CSP Explore Task Practice	IOC-2.A.2 IOC-2.A.10 IOC-2.A.3 IOC-2.A.14 IOC-2.A.4 IOC-1.F.11 IOC-2.A.5 CRD-1.A.1 IOC-2.A.6 CRD-1.A.2	Artifact Creation Computing Innovations Data Input and Output Data Privacy and Security

Prepare for Create PT	ALL	Review Course Content Incremental Development Documentation Debugging Collaborative Development
Create PT		12 hours of class time to conduct Create PT
Example Activity and Big Idea/Computational Thinking Practice *Create Performance Task:* Students develop a program of their choice. Their development process includes iteratively designing, implementing, and testing their program. Students are strongly encouraged to work with another student in their class. **[Big Idea AAP][Computational Thinking Practices 1-4]**		

Unit 17: Review for the AP Exam (1 week, 5 hours)
This unit gives students a review of the topics covered in the course and provides practice solving AP Exam style multiple-choice questions.

Subsection	**Lessons / Topics**
Prepare for Practice Exam	Review course content What to expect on the exam
Practice AP Exam	Cumulative Final AP Review Multiple Choice Test

Unit 18: Creative Development (Remainder of the school year, 2-4 weeks, 10-20 hours)
In this unit, students will brainstorm their own final project, discuss their ideas with their peers, scope their project to fit within the time constraints of the class, plan out milestones for incremental development, and create their own final product from scratch. This project allows students to think creatively about the applications of the concepts covered in the course, and create something of personal value.

Subsection	EKs	Lessons / Topics
Design Thinking Lessons: *Intro to Design Thinking*	CRD-1.A.4 CRD-2.E.4 CRD-1.A.5 CRD-2.F.1 CRD-1.A.6 CRD-2.F.2 CRD-2.A.1 CRD-2.F.5 CRD-2.A.2 CRD-2.F.6 CRD-2.E.1 CRD-2.F.7 CRD-2.E.2 IOC-1.A.2	Computing Innovations Development Process Program Specifications Design Phase Communication Collaboration
Brainstorm, Prototype & Test Lessons: *Prototype* *Test*	CRD-2.E.2 CRD-2.F.4 CRD-2.F.7 CRD-2.F.3 CRD-1.A.5 IOC-1.D.1 CRD-1.A.6 IOC-1.D.2 CRD-1.A.4 IOC-1.D.3 CRD-2.E.3 IOC-1.F.11	Development Process User Testing User Research Diverse Perspectives Iterative Development Human Biases Legal and Ethical Concerns
Project Prep and Development Lessons: *Project Prep and Development*	CRD-1.B.1	Online Collaboration Tools
Example Activity and Big Idea/Computational Thinking Practice *User Interface Scavenger Hunt:* Students search for 2 websites or apps, one with a good UI and one with a not-so-good UI. They learn to discriminate features of solid UI design in terms of accessibility and more before moving onto prototyping their creative project for the unit. **[Big Idea CRD][Computational Thinking Practices 6]**		

Supplemental Materials

Supplementary Units	**Prerequisite/Recommended Unit(s)**	**# of activities**
Extra Karel Practice	Intro to Programming	12
Extra Karel Puzzles	Intro to Programming	11
Karel Challenges	Intro to Programming	7
Web Development	After Pretest	79
Functions and Parameters Practice	Functions & Parameters	8

Extra Console Challenges - Prime Numbers	Javascript Control Structures	10
Animation and Games - Timers - Random Circles - Random Ghosts - Bouncing Ball - Mouse Events: Mouse Clicked - Mouse Events: Mouse Moved - Drawing Lines - Key Events - Crazy Ball Game	Functions & Parameters	51
Project: Breakout	Functions & Parameters	4
Data Structures Challenge Problems - Conway's Game of Life - Connect Four	Basic Data Structures	6
Visualizing Music	Basic Data Structures	9
Project: Tic Tac Toe	Basic Data Structures	4
Project: Helicopter	Basic Data Structures	24
More Basic Data Structures	Basic Data Structures	38