AP Computer Science Principles Course Syllabus

with Unplugged & Physical Computing Activities

Introduction

AP Computer Science Principles is the newest AP® course from the College Board, and becomes an official AP® course in the 2016-17 school year. This course introduces students to the foundational concepts of computer science and explores the impact computing and technology have on our society.

With a unique focus on creative problem solving and real-world applications, the CodeHS AP Computer Science Principles course gives students the opportunity to explore several important topics of computing using their own ideas and creativity, use the power of computing to create artifacts of personal value, and develop an interest in computer science that will foster further endeavors in the field.

Course Overview

Prerequisites: There are no official prerequisites for the CodeHS AP 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. However, we recommend that students take our Introduction to Computer Science prior to our AP courses (more info at https://codehs.com/course/catalog). Students who have completed our Intro to CS course will be able to apply knowledge of concepts covered in the Intro course to the more advanced setting of the AP courses. We also recommend that students complete a first-year high school algebra course prior to taking this course. Students should be comfortable with functions and function notation such as $f(x) = x + 2$ as well as using a Cartesian (x, y) coordinate system to represent points in a plane.

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, and students will use Processing.js to create graphical programs. They will also create webpages using HTML, CSS, and JavaScript. These webpages will be hosted on the CodeHS website so that they can keep a running portfolio of their creative projects, and easily share their programs with the world. 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.

Quizzes: At the end of each unit, students take a summative multiple choice unit quiz in the style of the AP Exam that assesses their knowledge of the concepts covered in the unit. Included in each lesson is a formative short quiz of various question types, including multiple choice, free response, and matching. The course also provides an AP Test Practice unit with a cumulative AP Practice Multiple Choice Test.

**** The Constellations Center believes that every student regardless of gender, race, ethnicity, or income should have access to quality computing education. Engaging students in different modalities of learning computer science is key to motivating interest. Additionally, making connections to solving problems students can relate to fosters their sense of exploration into how computer science affects their lives and their communities.*

Unplugged Activities: *Activities that introduces fundamental building blocks of computer science—without using computers at all. CS Unplugged is suitable for people of all ages, from elementary school to seniors, and from many countries and backgrounds.*

Physical Computing: *Instructional strategy that teaches students about computer science and computational thinking through physical tools and hands-on activity. Sometimes referred to as "maker spaces" or "project-based learning," physical computing is meant to encourage interdisciplinary and entrepreneurial thinking and foster student creativity.*

Course Objectives

This course is based directly off of the College Board AP Computer Science Principles Framework. We recommend reading the curriculum framework here for context. The main course objectives are summarized below in the six computational thinking practices and seven big ideas for the course.

Computational Thinking Practices: The six computational thinking practices represent important aspects of the work that computer scientists engage in, and are denoted here by P1 through P6:

- P1: Connecting Computing
    - Identify impacts of computing.
    - Describe connections between people and computing.

    - Explain connections between computing concepts.
- P2: Creating Computational Artifacts
    - Create an artifact with a practical, personal, or societal intent.
    - Select appropriate techniques to develop a computational artifact.
    - Use appropriate algorithmic and information management principles.
- P3: Abstracting
    - Explain how data, information, or knowledge is represented for computational use.
    - Explain how abstractions are used in computation or modeling.
    - Identify abstractions.
    - Describe modeling in a computational context.
- P4: Analyzing Problems and Artifacts
    - Evaluate a proposed solution to a problem.
    - Locate and correct errors.
    - Explain how an artifact functions.
    - Justify appropriateness and correctness of a solution, model, or artifact.

- P5: Communicating
    - Explain the meaning of a result in context.
    - Describe computation with accurate and precise language, notations, or visualizations.
    - Summarize the purpose of a computational artifact.
- P6: Collaborating
    - Collaborate with another student in solving a computational problem.
    - Collaborate with another student in producing an artifact.
    - Share the workload by providing individual contributions to an overall collaborative effort.
    - Foster a constructive, collaborative climate by resolving conflicts and facilitating the contributions of a partner or team member.
    - Exchange knowledge and feedback with a partner or team member.
    - Review and revise their work as needed to create a high-quality artifact.

Big Ideas: The seven big ideas of the course encompass foundational ideas of the field of computer science, and are denoted here by B1 through B7:

- B1: Creativity
    - How can a creative development process affect the creation of computational artifacts?

- How can computing and the use of computational tools foster creative expression?
- How can computing extend traditional forms of human expression and experience?

- B2: Abstraction
  - How are vastly different kinds of data, physical phenomena, and mathematical concepts represented on a computer?
  - How does abstraction help us in writing programs, creating computational artifacts, and solving problems?
  - How can computational models and simulations help generate new understanding and knowledge?
- B3: Data and Information
  - How can computation be employed to help people process data and information to gain insight and knowledge?
  - How can computation be employed to facilitate exploration and discovery when working with data?
  - What considerations and tradeoffs arise in the computational manipulation of data?
  - What opportunities do large data sets provide for solving problems and creating knowledge?

- B4: Algorithms
  - How are algorithms implemented and executed on computers and computational devices?
  - Why are some languages better than others when used to implement algorithms?
  - What kinds of problems are easy, what kinds are difficult, and what kinds are
  - impossible to solve algorithmically?
  - How are algorithms evaluated?

- B5: Programming
  - How are programs developed to help people, organizations, or society solve problems?
  - How are programs used for creative expression, to satisfy personal curiosity, or to create new knowledge?
  - How do computer programs implement algorithms?
  - How does abstraction make the development of computer programs possible?
  - How do people develop and test computer programs?
  - Which mathematical and logical concepts are fundamental to computer programming?
- B6: The Internet

- ○ What is the Internet? How is it built? 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?

- B7: Global Impact
  - ○ How does computing enhance human communication, interaction, and cognition?
  - ○ How does computing enable innovation?
  - ○ What are some potential beneficial and harmful effects of computing?

The AP Performance Tasks: The through course assessment is a set of performance tasks designed to gather evidence of student proficiency in the learning objectives. The AP Performance Tasks (PTs) are in-class assessments, administered by the teacher, that allow students to exemplify their learning through authentic, "real-world" creations. For more information about the AP Performance Tasks, refer to the curriculum framework.

The two performance tasks as defined by College Board are:

1. Explore - Implications of Computing Inventions
   - ○ Students explore the impacts of computing on social, economic, and cultural areas of our lives.
2. Create - Applications from Ideas
   - ○ Students create computational artifacts through the design and development of programs.

Students will gain the experience necessary to complete the PTs in class. Each unit comes with practice PTs in which students will research topics in computing, and create their own digital artifacts. Students will create and maintain a website that will hold each student creation throughout the course. This will serve as a running portfolio of each creative project the student completes. Sufficient time is set aside in the course for students to prepare for and complete both PTs.

The AP Exam: This course will prepare students for the multiple choice AP Computer Science Principles examination. Each lesson comes with quizzes to test essential knowledge for the AP Exam. Each unit includes a cumulative AP style multiple choice exam to test understanding of the concepts in the unit, and provide immediate feedback to the student.

Course Breakdown

Unit 1: Web Development (3 weeks)

In this unit, students will go through a high level introduction to HTML, CSS, and the processes involved in viewing web pages on the internet. Students will create several simple web pages using the CodeHS online editor to gain practice using the various features of HTML and CSS. This unit culminates with each student making their own website about themselves, hosted on their own custom CodeHS URL. This personal website will be continually improved by the student as they continue on in the course, and will serve as a running portfolio of each creative project they create in the course.

Subsection	EU	LO [P] (Ek)	Lessons / Topics
Getting Started	1.1 1.2 1.3 3.2 5.1 5.2 6.1 7.1 7.2	1.2.1 [P2] (A-E) 1.3.1 [P2] (A-E) 5.1.1 [P2] (A-F) 5.2.1 [P3] (K) 7.1.1 [P4] (A, H, L, O) 7.2.1 [P1] (A-D, F, G)	Why AP CSP? Impact of computing Innovations in computing What is HTML? Why is it important? What is a markdown language? How is HTML used? Sharing ideas and creations through web pages
HTML		5.1.1 [P2] (A-E) 5.1.2 [P2] (A-C) 3.2.1 [P1] (G, H)	Hierarchical layout of HTML documents HTML tags Formatting Lists Tables Creating your own websites
CSS		5.1.1 [P2] (A-E) 5.1.2 [P2] (A-C)	
Viewing Websites		6.1.1 [P3] (A-I) 7.1.1 [P4] (A-F, H, M-O)	Locating a resource with URLs Requesting a resource Browsers
Practice PT		1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 1.2.3 [P2] (A-C) 1.2.5 [P4] (A-D) 5.1.1 [P2] (A-E) 5.1.2 [P2] (A-C)	Practice Create PT - Make your own Homepage! Students will build their own website about themselves. This site will be accessible on their own custom url on the CodeHS site, and will be continually improved by the student as they continue on in the course. It will serve as a running portfolio of each creative project they create in the course.

The Web Development unit builds toward the following Enduring Understandings (EUs):
EU 1.1 Creative development can be an essential process for creating computational artifacts. (LO 1.1.1)

EU 1.2 Computing enables people to use creative development processes to create computational artifacts for creative expression or to solve a problem. (LO 1.2.1) EU 1.3 Computing can extend traditional forms of human expression and experience. (LO 1.3.1)

- To build toward EU 1.1, EU 1.2, and EU 1.3, Unit one includes a focus on beginning the development of a website with the use of HTML, an effective computational tool that

introduces students to simple structures of a web page. Unit one also encourages a creative process that will be continually refined over various points in the course. Students will have the opportunity to improve on their own website in an on-going and iterative manner.

EU 3.2 Computing facilitates exploration and the discovery of connections in information. (LO 3.2.1)

- Unit 1 builds toward EU 3.2 by introducing the concepts of HTML tables and lists focusing on the importance of structuring the use of data and information that is displayed on a web page in a way that is easy to read and comprehend.

EU 5.1 Programs can be developed for creative expression, to satisfy personal curiosity, to create new knowledge, or to solve problems (to help people, organizations, or society). (LOs 5.1.1, 5.1.2)

EU 5.2 People write programs to execute algorithms.

- Concepts in Unit 1 build toward EU 5.1 and EU 5.2 by focusing on the development of foundational habits of mind that they will be able to draw upon as they continue to iteratively enhance their own web page at various points in the course. The unit introduces the use of computational tools (HTML and CSS) commonly used to create a web page and as students learn simple commands to begin creating their web page, students are also learning the importance of algorithms in programs.

EU 6.1 The Internet is a network of autonomous systems. (LO 6.1.1)

EU 7.1 Computing enhances communication, interaction, and cognition. (LO 7.1.1)

- Unit 1 builds toward EU 6.1 and EU 7.1 by including the study of the how the viewing of a web page occurs on the Internet and it discusses the structures of the Internet that allow a web page to be accessible and visible to the world.

Unit 2: Introduction to Programming with Karel the Dog (4 weeks)

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 the 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. Unit 2: Introduction to Programming with Karel the Dog (4 weeks) 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 the 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	LO [P] (Ek)	Lessons / Topics
Unplugged Activity - Programming Languages		*Programming Languages*
Getting StartedStarted	2.2.3 [P3] (A-C) 5.1.1 [P2] (A-F) 5.2.1 [P3] (E, I, J, K)	What is a programming language? The need for programs How programs are developed How programs are used Abstractions present in programs and programming languages
Basic Karel Commands	4.1.1 [P2] (B) 5.2.1 [P3] (A, B, D)	Commands Sequential execution
Functions	1.2.5 [P4] (A-D) 2.2.1 [P2] (A) 5.3.1 [P3] (A-D)	Defining versus calling functions Designing functions Program entry points Control flow Debugging
Unplugged Activity - Towers of Hanoi		*Algorithms, sequencing*
Designing Algorithms	4.1.1 [P2] (B, E, H, I) 4.1.2 [P5] (A-C) 4.2.4 [P4] (A-F) 5.1.2 [P2] (A-C) 5.3.1 [P3] (A-D) 5.4.1 [P4] (D-F)	Preconditions and postconditions Top down design Pseudocode Comparing algorithms Debugging
Programming Style	1.2.4 [P6] (A-F) 1.2.5 [P4] (A-D) 4.1.2 [P5] (I) 5.1.2 [P2] (B-F) 5.1.3 [P6] (A-F) 5.4.1 [P4] (A, B, D-F, H-L, N)	The need for documentation Commenting code Preconditions and Postconditions Collaborating with others to solve computational problems
Control Structures	2.2.1 [P2] (A, B) 4.1.1 [P2] (A-F, H, I) 5.1.3 [P6] (A-F) 5.3.1 [P3] (A-D) 5.4.1 [P4] (B, D-G)	Looping Conditionals / Selection Generalizing solutions Writing reusable code

Practice PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 1.2.3 [P2] (A-C) 4.1.1 [P2] (A-E) 5.1.2 [P2] (A-C)	Practice Create PT: Write a Karel Program to Draw a Digital Image 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.

Units 3-5: Programming with JavaScript (6 weeks)

This unit introduces students to the basics of JavaScript, and gives students practice writing JavaScript programs to solve general problems. Students will be able to compare and contrast JavaScript with Karel and identify the abstractions Karel provides over JavaScript. Other topics in this unit include data structures, APIs, the importance of programming style, and the impact programming has had on the types of problems that can be solved. Students will use their knowledge of JavaScript to write a graphical program that tells a story, and embed this program in their portfolio website.

Subsection	LO [P] (Ek)	Lessons / Topics
Unit 3		
Getting Started	5.1.1 [P2] (A-F) 4.1.2 [P5] (A-H)	Why use different programming languages? JavaScript in the real world
The Basics	5.2.1 [P3] (A-E) 5.3.1 [P3] (J) 5.4.1 [P4] (C) 5.5.1 [P1] (A, D)	Printing Variables Types Arithmetic Expressions Input/Output
Unplugged Activity - Drawing Lines with Pixels		*Graphics*
Intro to Graphics	2.2.2 [P3] (A, B) 5.3.1 [P3] (M, N)	Making graphical programs with JavaScript
Boolean Logic	4.1.1 [P2] (C) 5.5.1 [P1] (E-G)	Booleans Boolean operators
Unit 4		
Control Structures	1.2.5 [P4] (A-D) 4.1.1 [P2] (A-F, H, I) 5.4.1 [P4] (A-N)	Loops Conditional statements Nested control structures Errors Debugging
Physical Computing Activity		*Sequencing, Selection, Iteration*

Traffic Light		
Unit 5		
Unplugged Activity - Functional Suncatchers		
Functions and Parameters	1.2.5 [P4] (A-D) 2.2.1 [P2] (A - C) 4.1.1 [P2] (A-H) 4.1.2 [P5] (G, I) 5.1.2 [P2] (A-J) 5.3.1 [P3] (A-G) 5.4.1 [P4] (A-N)	Functions Parameters Return values Top down design Procedural abstraction Generalizing procedures Writing reusable code
Practice PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 1.2.3 [P2] (A-C) 5.4.1 [P4] (I-N)	Practice Create PT: Create a web comic Students will write a JavaScript program that draws a graphical comic strip to tell a story. They will then embed this web comic in their personal portfolio website.
Unit 6		
Physical Computing: Tank Game		*Scratch Game that encompasses functions, loops, and the HummingBird Kit*
Basic Data Structures	4.1.1 [P2] (A-I) 4.2.4 [P4] (A-F) 5.3.1 [P3] (H-L) 5.4.1 [P4] (A-N) 5.5.1 [P1] (H-J)	Data abstraction Arrays JavaScript Objects Iterating over data structures
Under the Hood	2.1.1 [P3] (A) 2.2.3 [P3] (A-J) 4.1.2 [P5] (C, E) 5.2.1 [P3] (A, E-H, K)	Computer hardware Abstractions present in writing and running programs CPU Memory Chips Logic gates Input / Output
Reflection	2.2.2 [P3] (A, B) 2.2.3 [P3] (A, B, D) 4.1.2 [P5] (A-I) 5.1.1 [P2] (A-D)	Comparing JavaScript to Karel Identifying abstractions present in Karel The impact of computer programming
Simulations	1.2.5 [P4] (A-D) 1.3.1 [P2] (E) 2.2.2 [P3] (A, B) 2.3.1 [P3] (A-D) 2.3.2 [P3] (A-H) 4.1.1 [P2] (E, G) 5.4.1 [P4] (A-N)	Modeling and simulating real world scenarios Random numbers Simplifying assumptions Forming hypotheses Modeling coin flips Conway's Game of Life

Practice PT	1.2.1 [P2] (A,B) 1.2.2 [P2] (A, B) 1.2.4 [P6] (A-F) 2.2.2 [P3] (A, B) 2.3.1 [P3] (A-D) 2.3.2 [P3] (A-H) 3.1.2 [P6] (B, D) 5.1.2 [P2] (A-J) 5.1.3 [P6] (A-F) 5.4.1 [P4] (I-N)	Practice Create PT: Simulating the World Students will work in groups to write a program that simulates a phenomenon of interest based on both simple and complex models. They will answer questions to describe their simulation as well as their development process. This simulation will be embedded in their personal portfolio website.

Unit 7: Digital Information (6 weeks)

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	LO [P] (Ek)	Lessons / Topics
Getting Started	2.1.1 [P3] (A) 3.1.1 [P4] (A)	The need for digital information Data abstractions Real world encodings Non computational encodings
Unplugged Activity - Count the Dots Online Extensions - Binary Game		*Intro into Binary* *Binary Game (online)*
Number Systems	2.1.1 [P3] (D, G)	The need for number systems How number systems work Decimal Binary Converting between number bases
Encoding Data with Binary	2.1.1 [P3] (A-E) 2.1.2 [P5] (A-F) 2.2.3 [P3] (K) 3.2.1 [P1] (G, H) 5.3.1 [P3] (H, J) 5.5.1 [P1] (A-C)	Why is data stored in binary? How is data stored in binary? Encoding numbers Encoding text Standard text encoding Custom text encoding Limits of encoding schemes
Encoding Black and White Images	1.2.3 [P2] (A-C) 1.3.1 [P2] (C) 2.1.1 [P3] (A-E) 2.1.2 [P5] (D, F) 2.2.3 [P3] (K) 3.2.1 [P1] (G, H)	Encoding images with pixels Representing black and white images with binary pixels Programmatically creating black and white images

Hexadecimal	2.1.1 [P3] (F, G)	The uses of the hexadecimal number system Converting between hexadecimal, decimal, and binary
Encoding Color Images	1.2.3 [P2] (A-C) 1.3.1 [P2] (C, D) 2.1.1 [P3] (A-G) 2.1.2 [P5] (D, F) 2.2.3 [P3] (K) 3.2.1 [P1] (G, H) 5.3.1 [P3] (H, M-O)	Color pixels The RGB color scheme Custom color schemes Trade-offs between data representations Programmatically modifying color images using a web API
Unplugged Activity - You Can Say That Again (Text Compression)		*Introduction into data compression*
Data Compression	2.1.2 [P5] (D, F) 3.2.1 [P1] (G, H) 3.3.1 [P4] (A, C-E, G)	The need for data compression Comparing trade offs of compression algorithms Lossless compression Lossy compression File formats
Unplugged Activity - Kid Krypto		*Intro into Encryption and it's key to data security.*
Cryptography	3.3.1 [P4] (A, B, F) 4.2.1 [P1] (A-C) 4.2.2 [P1] (D) 4.2.3 [P1] (A-C) 6.3.1 [P1] (H-L)	The need for encryption Symmetric encryption Public key encryption Historic encryptions Caesar Ciphers Steganography The computability of problems "Hard" vs "Easy" problems Unsolvable problems Why certain encryptions are "hard" to decrypt
Practice PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-D) 1.2.4 [P6] (A-F) 1.2.5 [P4] (A-D) 2.1.2 [P5] (D-F) 2.2.1 [P2] (A-C) 3.1.2 [P6] (B) 5.4.1 [P4] (I-N)	Practice Create PT: Develop your own image filter 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.

Unit 8: The Internet (6 weeks)

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 in 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	LO [P] (Ek)	Lessons / Topics
Getting Started	3.1.2 [P6] (B, E) 6.1.1 [P3] (A, D) 7.1.1 [P4] (A-F, H, I, M-O) 7.1.2 [P4] (A-G) 7.2.1 [P4] (C-E, G) 7.3.1 [P4] (A-G, I, L, N)	What is the Internet Impacts of the Internet Legal and ethical concerns Open and collaborative nature of the Internet Reflecting on the Internet's role in your life
Unplugged Activity - The Orange Game		*Intro into data transmission and error detection*
Internet Hardware	2.1.1 [P3] (E) 6.1.1 [P3] (A) 6.2.1 [P5] (A) 6.2.2 [P4] (I-K)	Networks Routers Sending bits at a physical level Abstractions in Internet connections
Internet Addresses and DNS	6.1.1 [P3] (A-I) 6.2.1 [P5] (A-C) 6.2.2 [P4] (A, C-E)	The need for addressing IP DNS Hierarchical layout of IP and DNS Scaling the Internet Collaborative development and oversight of the Internet
Routing	4.2.1 [P1] (D) 4.2.2 [P1] (A-C) 6.2.1 [P5] (D) 6.2.2 [P4] (A, B, D)	Routing Redundancy Reliability Routing algorithms Heuristics to find the shortest path
Packets and Protocols	6.2.1 [P5] (A) 6.2.2 [P4] (A, D-H)	The need for packets TCP/IP HTTP Layers of abstraction in the Internet
Unplugged Activity - Wireless Hotspot Problem		*Intro into internet availability, equity vs access.*
Impact of the Internet	1.2.3 [P2] (A-C) 6.1.1 [P3] (A, D) 7.1.1 [P4] (A-F, H, I, M-O) 7.3.1 [P4] (A-	The positive and negative impacts of the Internet on society Access to information Privacy Intellectual Property and Copyright Censorship Anonymity Internet Reliance The Digital Divide

	G, I, L, N-Q) 7.4.1 [P1] (A-E)	
Cybersecurity	6.3.1 [P1] (A-H, M) 7.3.1 [P4] (G)	Problems in cybersecurity Trust model of the Internet Cybersecurity developments Digital certificates
Practice PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A, B, D, E) 1.2.2 [P2] (A, B) 6.1.1 [P3] (A, D) 7.1.1 [P4] (A-F, H, I, M-O) 7.5.1 [P1] (A-C) 7.5.2 [P5] (A, B)	Practice Explore PT: The Effect of the Internet on Society 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.

Unit 9: Data (5 weeks)

In this unit, students will explore using computational tools to store massive amounts of data, manipulate and visualize data, find patterns in data, and pull 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	LO [P] (Ek)	Lessons / Topics
Getting Started	3.1.1 [P4] (A, D, E) 3.2.1 [P1] (A-F) 3.2.2 [P3] (A, B, D, G)	The modern wealth of data Insights gained from data
Interpreting Data	3.1.2 [P6] (A-F) 3.2.1 [P1] (G-I)	Gaining insights from exploring large data sets Metadata Limitations to what can be concluded from data
Visualizing Data	3.1.3 [P6] (A-E) 7.1.1 [P4] (F, G)	Gaining insights from data visualizations Comparing visualizations Producing visualizations
Collecting Data	3.1.1 [P4] (A, B) 3.1.2 [P6] (A-E) 3.2.2 [P3] (B, C, E, H) 3.3.1 [P4] (A, B,	Ways to store data Structuring data sets for analysis Trade-offs between data storage schemas

	F-I) 7.1.1 [P4] (J, K) 7.1.2 [P4] (F, G)	
Unplugged Activity - 20 Questions		*Intro into binary and linear searching.*
Programmatically Manipulating Data	3.1.1 [P4] (A-E) 3.1.3 [P6] (A-E) 4.2.4 [P4] (A-H) 5.3.1 [P3] (J-O)	Searching algorithms Sorting algorithms Comparing algorithms Programmatically accessing data APIs Using computational tools to find patterns in data
Impact of Data Collection and Analysis	3.2.1 [P1] (A-I) 3.1.1 [P4] (A, D, E) 3.2.2 [P3] (A-H) 7.1.1 [P4] (J, K) 7.1.2 [P4] (F, G) 7.3.1 [P4] (H-K, M)	Predicting with data models Privacy rights Societal impacts of data collection and analysis
Practice PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 1.2.2 [P2] (A, B) 1.2.4 [P6] (A-F) 3.1.2 [P6] (A-F) 3.1.3 [P6] (A-E) 3.2.1 [P1] (A-I) 7.1.1 [P4] (E-H)	Practice Explore PT: Present a Data-Driven Insight 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.

Unit 10: Performance Tasks (4 weeks)

This time is set aside for students to prepare for and create their AP Performance Tasks. Students will be given the chance to review course content and practice the skills necessary to complete each performance task. The Explore PT will be administered over 8 hours of class time, and the Create PT will be administered over 12 hours of class time.

Subsection	LO [P] (Ek)	Lessons / Topics
Prepare for Explore PT	1.2.1 [P2] (A, B) 7.5.1 [P1] (A - C) 7.5.2 [P5] (A, B) 3.2.1 [P1] (D)	Review course content Conducting efficient research Effective time management
Explore PT		8 hours of class time to conduct Explore PT
Prepare for Create PT	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 5.1.1 [P2] (A, B, C)	Review course content Project scoping Incremental development Documentation

	5.1.2 [P2] (A-F, J) 5.1.3 [P6] (A-F)	Debugging Collaborative program development
Create PT		12 hours of class time to conduct Create PT

Unit 11: Review for the AP Exam (1 week)

This unit gives students a review of the topics covered in the course and provides practice solving AP Exam style multiple choice questions.

Subsection	LO [P] (Ek)	Lessons / Topics
Prepare for Practice Exam	(Sample of entire course)	Review course content What to expect on the exam
Practice AP Exam	(Sample of entire course)	Cumulative Final AP Review Multiple Choice Test with immediate feedback

Unit 12: Final Project (Remainder of the school year)

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	LO [P] (Ek)	Lessons / Topics
Brainstorm	1.1.1 [P2] (A, B) 1.2.1 [P2] (A-E) 1.2.2 [P2] (A, B) 1.2.3 [P2] (A-C) 1.2.4	Brainstorm ideas for a final project Allow students to think creatively about the applications of the concepts covered in the course Discuss project ideas with peers

	[P6] (A-F)	
Project Planning	1.1.1 [P2] (A, B) 5.1.1 [P2] (A-C) 5.1.2 [P2] (A-J)	Project scoping Incremental development Plan out project milestones
Hummingbird Project		*https://www.birdbraintechnologies.com/hummingbirdduo/teach/*
Software Design	1.1.1 [P2] (A, B) 1.2.4 [P6] (A-F) 1.2.5 [P4] (A-D) 5.1.2 [P2] (A-J) 5.3.1 [P3] (M-O)	Designing an application from scratch Evaluate and compare proposed solutions Utilizing open source software
Implementation	1.1.1 [P2] (A, B) 1.2.4 [P6] (A-F) 1.2.5 [P4] (A-D) 5.1.1 [P2] (A-C) 5.1.2 [P2] (A-J) 5.1.3	Create your final product Collaborative program development Testing Debugging

	[P6] (A-F)	
Final Demo and Presentation	5.4.1 [P4] (G, I-N)	Communicate the value of the final product via product demo, infographic, video, or a computational artifact of your choosing