AP Computer Science Principles In Roblox Course Syllabus

High School (175 Contact Hours)

Introduction

AP Computer Science Principles 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 in Roblox 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.

The CodeHS AP Computer Science Principles in Roblox course utilizes Roblox Studio and the Lua/Luau programming language to teach students programming concepts. Students will build worlds and games in Roblox, and they will be encouraged to use Roblox for their Performance Task assessment. In addition, they will explore computer science concepts through Roblox simulations.

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 an Introduction to Computer Science course prior to our AP courses (more info at codehs.com/library). Students who have completed an 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.

Overarching Course Goals:

- Increase and diversify participation in computer science
- Students, regardless of prior experience in computing, will develop confidence using computer science as a tool to express themselves and solve problems, and this confidence will prepare them for success in future endeavors in the field of computer science
- Students will understand the core principles of computing, a field that has and continues to change the world

- Students will be able to develop computational artifacts to solve problems, communicate ideas, and express their own creativity
- Students will be able to collaborate with others to solve problems and develop computational artifacts
- Students will be able to explain the impact computing has on society, economy, and culture
- Students will be able to analyze existing artifacts, identify and correct errors, and explain how the artifact functions
- Students will be able to explain how data, information, or knowledge is represented for computational use
- Students will be able to explain how abstractions are used in computation and modeling
- Students will learn to be informed and responsible users of technology

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 and in Roblox Studio. They will create 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 module of the course is broken down into lessons. Lessons consist of video tutorials, 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 and the third-party development environment Roblox Studio. Students will be able to write text-based programs in the Lua programming language utilized in Roblox, and students will use Roblox Studio to create graphical programs, games, objects, and worlds. 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.

Technical Requirements: Access to a computer and high-speed internet is required. Refer to this FAQ article to see the full list of requirements.

Quizzes**:** At the end of each module, 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. The course also provides an AP Test Practice unit with a cumulative AP Practice Multiple Choice Test.

Course Objectives

This course is based directly on 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 five 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:

- **Practice 1: Computational Solution Design**
  - *Design and evaluate computational solutions for a purpose.*
- **Practice P2: Algorithms and Program Development**
  - *Develop and implement algorithms.*
- **Practice P3: Abstraction in Program Development**
  - *Develop programs that incorporate abstractions.*
- **Practice P4: Code Analysis**
  - *Evaluate and test algorithms and programs.*
- **Practice P5: Computing Innovations**
  - *Investigate computing innovations.*
- **Practice P6: Responsible Computing**
  - *Contribute to an inclusive, safe, collaborative, and ethical computing culture.*

Big Ideas:

The five big ideas of the course encompass foundational ideas in the field of computer science, and are denoted here by B1 through B5:

- **Big Idea 1: Creative Development (CRD)**
  When developing computing innovations, developers can use a formal, iterative design process or experimentation. While using either approach, developers will encounter phases of investigating and reflecting, designing, prototyping, and testing. Additionally, collaboration is an important tool to use at any phase of development because considering multiple perspectives allows for improvement of innovations.

- **Big Idea 2: Data (DAT)**
  Data is central to computing innovations because it communicates initial conditions to programs and represents new knowledge. Computers consume data, transform data, and produce new data, allowing users to create new information or knowledge to solve problems through the interpretation of this data. Computers store data digitally, which means that the data must be manipulated in order to be presented in a useful way to the user.

- **Big Idea 3: Algorithms and Programming (AAP)**
  Programmers integrate algorithms and abstraction to create programs for creative purposes and to solve problems. Using multiple program statements in a specified order, making decisions, and repeating the same process multiple times are the building blocks of programs. Incorporating elements of abstraction, by breaking problems down into interacting pieces, each with their own purpose, makes writing complex programs easier. Programmers need to think algorithmically and use abstraction to define and interpret processes that are used in a program.

- **Big Idea 4: Computing Systems and Networks (CSN)**
  Computer systems and networks are used to transfer data. One of the largest and most commonly used networks is the Internet. Through a series of protocols, the Internet can be used to send and receive information and ideas throughout the world. Transferring and processing information can be slow when done on a single computer but leveraging multiple computers to do the work at the same time can significantly shorten the time it takes to complete tasks or solve problems.

- **Big Idea 5: Impact of Computing (IOC)**
  Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand how our programs will be used and be responsible for the consequences. As computer users, we need to understand how to protect ourselves and our privacy when using a computer.

The AP Create Performance Task:

The through course assessment is a performance task designed to gather evidence of student proficiency in the learning objectives. The AP Create Performance Tasks (PT) is an in-class assessment, administered by the teacher, that allows students to exemplify their learning through an authentic, "real-world" creation. In the Create Performance Task, students will design and implement a program to solve a problem, enable innovation, explore personal interest, or express creativity. Their development process should include exploration, investigation, reflection, design, implementation, and testing your program.

Students will gain the experience necessary to complete the Create Performance Task in class. Each unit comes with practice PTs in which students will research topics in computing, and create their own digital artifacts. Sufficient time is set aside in the course for students to prepare for and complete the Create Performance Task.

The AP Exam:

The AP Computer Science Principles end-of-course exam has consistent question types and weighting every year, so you and your students know what to expect on exam day.

Section I: End-of-Course Multiple-Choice Exam
70 multiple-choice questions | 120 minutes | 70% of score | 4 answer options

- 57 single-select multiple-choice
- 5 single-select with reading passage about a computing innovation
- 8 multiple-select multiple-choice: select 2 answers

Section II: Create Performance Task: Written Responses
30% of score

- Create Performance Task program code, video, and student-authored Personalized Project Reference | 9 hours in-class
- 4 written response prompts | 60 minutes end-of-course exam

The second section of the AP Computer Science Principles Exam consists of a through-course Create Performance Task where students will develop a computer program of their choice and an end-of-course written response section where students demonstrate their understanding of their personal Create Performance Task by answering four prompts. Students will be provided 9 hours of in-class time to complete their program, video, and develop a Personalized Project Reference.

Course Breakdown

Module 1: Getting Started with Roblox Studio (3 weeks, 15 hours)

In this module, students will learn the basics of programming while becoming familiar with coding in Roblox Studio. The basic lesson structure will include an introductory programming concept in the CodeHS editor. Then, students will apply this concept to programming in Roblox Studio.

Subsection	EKs	Lessons / Topics
Programming Languages Lessons: *What is Computer Science?*	AAP-2.A.2 AAP-2.A.3 CRD-1.A.1 CRD-1.A.2 CRD-2.B.1	What is Programming? Pseudocode Programming Languages Computing Innovations
Abstraction Lessons: *Abstraction and APIs*	AAP-3.B.1 AAP-3.B.2 AAP-3.B.3 AAP-3.B.4 AAP-3.B.6 AAP-3.B.7 CRD-2.G.1 DAT-1.A.2 DAT-1.A.5	Procedural Abstraction Modularity Program Reuse Digital Data (Bits) Reducing Complexity
Programming Style Lessons: *Intro to Programming with Lua* *Abstraction and APIs*	CRD-2.G.1 CRD-2.G.2 AAP-2.M.1 AAP-2.M.3 CRD-2.B.1 CRD-2.B.2 CRD-2.B.5 AAP-3.D.1 AAP-3.D.2 AAP-3.D.3 AAP-3.D.4 AAP-3.D.5	Program Documentation Using Existing Code and Libraries APIs Commenting Code
Debugging Strategies Lessons: *Debugging Strategies in Roblox Studio*	CRD-2.I.1 CRD-2.I.2 CRD-2.I.3 CRD-2.I.5	Logic Errors Syntax Errors Run-Time Error Testing
Variables Lessons: *Variables*	AAP-1.A.1 AAP-1.A.2 AAP-1.A.3 AAP-1.A.4 AAP-1.B.1 AAP-1.B.2 AAP-1.B.3 DAT-1.A.1	Variable Names Assignment Operators Data Types Variables as Abstractions

Arithmetic Expressions Lessons: *Basic Math in Lua* *Introduction to Programming with Lua*	CRD-2.B.4 CRD-2.I.5 CRD-2.J.1 CRD-2.J.2 CRD-2.J.3 AAP-2.A.1 AAP-2.A.2 AAP-2.A.3 AAP-2.A.4 AAP-2.B.3 AAP-2.B.4 AAP-2.B.5 AAP-2.C.1 AAP-2.C.2 AAP-2.C.3 AAP-2.C.4 AAP-2.D.1 AAP-2.D.2	Program Behavior Testing using Inputs Arithmetic Expressions Order of Operations Modulus String Concatenation

Module 2: User Interaction and Control Structures (2.5 weeks, 13 hours)

In this module, students extend their understanding of programming in Lua to use control structures to create more complex programs. Students also learn about the Humanoid object and how to use Touched Events to make their programs interactive.

Subsection	EKs	Lessons / Topics
Control Structures Lessons: *Booleans and Conditionals* *For Loops* *While Loops*	AAP-2.G.1 AAP-2.J.1 AAP-2.K.1	If/Else Statements (Selection) For Loops and While Loops (Iteration)
Comparison Operators Lessons: *Comparison and Logical Operators* *Operators in Roblox*	AAP-2.E.1 AAP-2.F.4 AAP-2.E.2 AAP-2.F.1 AAP-2.F.2 AAP-2.F.3 AAP-2.F.5	Booleans Relational Operators Operands
Selection Lessons: *Booleans and Conditionals* *Random Numbers*	AAP-2.G.1 AAP-2.H.1 AAP-2.L.3 AAP-2.H.2 AAP-2.H.3 AAP-2.I.1 AAP-2.I.2 AAP-2.L.4 AAP-3.E.2	Selection Conditional Statements Nested Conditionals Equivalent Boolean Statements Random Numbers
Iteration Lessons: *While Loops* *For Loops*	AAP-2.K.2 AAP-2.K.3 AAP-2.L.2 AAP-2.K.4 AAP-2.K.5 AAP-2.L.1 AAP-2.L.5	Iteration Loops Different but Equivalent Algorithms
Designing Algorithms	AAP-2.A.4	Sequencing, Selection, Iteration

Lessons: *Algorithms*	AAP-2.M.2 AAP-2.B.1 AAP-4.A.2 AAP-2.B.2 AAP-4.A.4 AAP-2.B.6 AAP-4.A.5 AAP-2.B.7 AAP-4.A.6	Clarity and Readability Using Existing Algorithms Optimization and Efficiency
Algorithm Efficiency Lessons: *Algorithms*	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

Module 3: Practice PT: Pair-Programming Obby! (3 days, 3 hours)

In this practice performance task, students will practice and use pair programming to create an obby (obstacle course) that utilizes at least six parts, a script, a function, a loop, variables, and a conditional statement.

Subsection	EKs	Lessons / Topics
Collaboration and Communication	CRD-1.A.3 CRD-1.A.4 CRD-1.B.2 CRD-1.C.1 CRD-2.F.5 CRD-2.F.6 CRD-2.F.7 CRD-2.G.1 CRD-2.G.3 CRD-2.G.4 CRD-2.G.5 CRD-2.H.1 CRD-2.H.2	Collaboration Diverse Perspectives Bias Avoidance Pair-Programming Design and Planning Program Documentation Acknowledgement of Reused Code

Module 4: Parameters and Return Values (2 weeks, 10 hours)

In this module, students learn how to write reusable code with functions and parameters.

Subsection	EKs	Lessons / Topics

User Input Lessons: *User Input*	AAP-1.C.4 AAP-3.A.6 AAP-3.A.9 CRD-2.C.2 CRD-2.C.3 CRD-2.C.5 CRD-2.C.6 CRD-2.D.2	User Input Adding Touched Events to Parts
Functions and Parameters Lessons: *Parameters* *Parameters Practice* *Return Values* *Return Values in Roblox*	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

Module 5: Practice PT: Scavenger Hunt (3 days, 3 hours)

In this practice performance task, students will create a scavenger hunt in Roblox. They will work on creating milestones and using pseudocode to program game instructions that direct the user to find items hidden in the world. They will iterate and test their code along the way to make sure they have solved each milestone. [**Big Idea CRD**][**Computational Thinking Practice 4**]

Module 6: Data Structures (2.5 weeks, 13 hours)

In this module, students learn about arrays as a way to store data more efficiently in their programs. They learn how to loop through arrays to access elements within an array and how to use folders to organize parts in Roblox Studio.

Subsection	EKs	Lessons / Topics
Basic Data Structures Lessons: *Intro to Arrays and Accessing an Element in an Array* *Adding and Removing Elements from an Array*	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 Arrays and Elements Indices Array Procedures
Data Abstractions	AAP-1.D.1	Data Abstraction

Lessons: *Looping Through Arrays* *Finding an Element in an Array*	AAP-1.D.5 DAT-2.E.4 AAP-1.D.2 AAP-1.D.3 AAP-1.D.4 DAT-2.E.2 DAT-2.D.4 DAT-2.E.5	Translating and Transforming Data Filtering and Cleaning Patterns
Traversing a List Lessons: *Looping Through Arrays* *Adding and Removing Elements from an Array*	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
Simulation Lessons: *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

Module 7: Digital Information (3 weeks, 15 hours)

In this module, students learn about the various ways to represent information digitally including number systems, encoding data, programmatically creating pixel images, comparing data encodings, and compressing and encrypting data.

Subsection	EKs	Lessons / Topics
Number Systems Lessons: *Intro to Digital Information* *Number Systems*	CRD-2.C.1 CRD-2.D.1 CRD-2.J.2 CRD-2.J.3 CRD-2.I.4 DAT-1.A.2 DAT-1.A.3 DAT-1.A.4 DAT-1.A.7 DAT-1.B.1 DAT-1.B.2 DAT-1.B.3 DAT-1.C.1 DAT-1.C.2 DAT-1.C.3 DAT-1.C.4	Computing Devices Abstraction Program Input and Output Bits and Bytes Overflow Errors Range of Value Limits Binary and Decimal Systems

	DAT-1.A.5 DAT-1.C.5 DAT-1.A.6	
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
Roblox Simulation Activity: CodeHS Escape Rooms This simulation is a single-player game that takes place in Karel's castle. Students are lost in the castle and need to find their way out of multiple rooms in order to explore the remainder of the castle. In the first room, students convert a randomly selected number in decimal format to a binary sequence using torches. Students discover RGB values hidden in the second room in order to add some color to the large banner hanging in the room. Correctly solving the puzzles in each room will open doors to other locations in Karel's castle.		

Module 8: Practice PT: Create a Color Filter (3 days, 3 hours)

In this practice performance task, students will create two custom color filters in Roblox studio. The player will be able to apply a color filter to a "pixel image" by pressing the button for that filter. Students can choose this practice PT or the previous one. **[Big Idea CRD][Computational Thinking Practice 1]**

Module 9: Practice PT: Steganography (3 days, 3 hours)

In this practice performance task, students will encrypt a secret message within the color code of Roblox parts! The player will be able to decrypt the message by clicking on each part. Students can choose this practice PT or the previous one. **[Big Idea CRD][Computational Thinking Practice 1]**

Module 10: The Internet (2 weeks, 10 hours)

In this module, students explore 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.

Subsection	EKs	Lessons / Topics

Internet Hardware and Addresses Lessons: *Welcome to the Internet* *Internet Hardware* *Internet Addresses*	CSN-1.A.1 CSN-1.A.2 CSN-1.A.3 CSN-1.A.4 CSN-1.A.7 CSN-1.A.8 CSN-1.B.3 CSN-1.B.4	Protocols Computing Devices Computer Networks Bandwidth
Routing Lessons: *Routing*	CSN-1.A.5 CSN-1.A.6 CSN-1.B.5 CSN-1.B.6 CSN-1.B.7 CSN-1.E.1 CSN-1.E.2 CSN-1.E.3 CSN-1.E.4 CSN-1.E.5 CSN-1.E.6 CSN-1.E.7	Routing Scalability Fault-Tolerance Redundancy
Packets and Protocols Lessons: *Packets and Protocols*	CSN-1.B.1 CSN-1.B.2 CSN-1.C.1 CSN-1.C.2 CSN-1.C.3 CSN-1.C.4 CSN-1.D.1 CSN-1.D.2 CSN-1.D.3 DAT-2.B.1 DAT-2.B.3 DAT-2.B.5	Datastreams Packets IP, TCP, UDP HTTP Metadata
Computing Systems Lessons: *Sequential, Parallel & Distributed*	DAT-2.C.7 DAT-2.C.8 CSN-2.A.1 CSN-2.A.2 CSN-2.A.3 CSN-2.A.4 CSN-2.A.5 CSN-2.A.6 CSN-2.A.7 CSN-2.B.1 CSN-2.B.2 CSN-2.B.3 CSN-2.B.4 CSN-2.B.5	Parallel Systems Scalability of Systems Sequential Computing Parallel Computing Distributed Computing Efficiency of Solutions Speedup
Impact of the Internet Lessons: *The Impact of the Internet* *Creative Credit and Copyright*	IOC-1.A.1 IOC-1.A.3 IOC-1.A.4 IOC-1.A.5 IOC-1.B.1 IOC-1.B.2 IOC-1.B.3 IOC-1.B.4 IOC-1.B.5 IOC-1.B.6 IOC-1.C.1 IOC-1.C.2 IOC-1.C.3 IOC-1.C.4 IOC-1.C.5 IOC-1.E.1 IOC-1.E.2 IOC-1.E.3 IOC-1.E.4 IOC-1.E.5 IOC-1.E.6 IOC-1.F.1 IOC-1.F.2 IOC-1.F.3 IOC-1.F.4 IOC-1.F.5 IOC-1.F.6 IOC-1.F.7 IOC-1.F.9 IOC-1.F.10 IOC-1.F.11	Computing Innovations Unintended Effects Impact on Society Rapid Sharing Digital Divide Citizen Science Crowdsourcing Creative Credit and Copyright
Cybersecurity	IOC-1.F.8 IOC-2.A.1 IOC-2.B.5 IOC-2.B.6	Legal and Ethical Concerns Personally Identifiable Info (PII)

Lessons: *Cybersecurity*	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	Digital Footprint Authentication Certificate Authorities (CAs) Computer Viruses Malware Phishing Keylogging Rogue Access Points Encryption
Roblox Simulation Activity: Routing Ruckus In this game, students must navigate between cities and towns while connecting network cables to floating hubs. Successfully completing the network with a high level of fault tolerance will award the student with a Roblox Network Navigator badge. This experience allows up to two players in each simulation at a time. Both players can work together to complete the network, but watch out for the UFO!		

Module 11: Project: 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, function, or effect. **[Computing Innovation 3, Prompt C][Big Idea IOC][Computational Thinking Practice 5]**

Module 12: Data (1 week, 5 hours)

In this module, 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.

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 DAT-2.B.1 DAT-2.C.4	Metadata Correlation Using a Variety of Sources

Lessons: *Data Collection and Limitations*	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	Incomplete or Invalid Data Bias Surveys, Testing, Interviews

Module 13: Project: 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, and communicate their findings to their classmates. **[Big Idea DAT][Computational Thinking Practice 6]**

Module 14: Project: Impact of Computing (3 days, 3 hours)

In this project module, students will explore computing innovations, reflect on how data can be collected and used, and consider the privacy and security concerns when person information is collected.

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

Modules 15 & 16: Create Performance Task and AP Exam Review (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 9 hours of class time.

Module 17: Creative Development (2-4 weeks, 10-20 hours)

In this module, 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-1.A.5 CRD-1.A.6 CRD-2.A.1 CRD-2.A.2 CRD-2.E.1 CRD-2.E.2 CRD-2.E.4 CRD-2.F.1 CRD-2.F.2 CRD-2.F.5 CRD-2.F.6 CRD-2.F.7 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.7 CRD-1.A.5 CRD-1.A.6 CRD-1.A.4 CRD-2.E.3 CRD-2.F.4 CRD-2.F.3 IOC-1.D.1 IOC-1.D.2 IOC-1.D.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