CS50 for AP Computer Science Principles

COURSE SYLLABUS – TABLE OF CONTENTS

CURRICULAR REQUIREMENTS

CR1a Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.

CR1b Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.

CR1c Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.

CR1d Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.

CR1e Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.

CR1f Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.

CR2a Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2b Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2c Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2d Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2e Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2f Students are provided with opportunities to meet learning objectives within Big Idea 6: The Internet. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR2g Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

CR3 Students are provided with the required amount of class time to complete the AP Through-Course Assessment *Explore – Impact of Computing Innovations* Performance Task.

CR4 Students are provided with the required amount of class time to complete the AP Through-Course Assessment *Create – Applications from Ideas* Performance Task.

CS50 for AP Computer Science Principles

COURSE SYLLABUS

COURSE INTRODUCTION

CS50 is Harvard University's introduction to the intellectual enterprises of computer science and the art of programming for students less comfortable and more comfortable alike. CS50 for AP Computer Science Principles is an adaptation of CS50 for high schools that aligns with the AP Computer Science Principles curriculum framework. The course assumes no prior background of students, but it is rigorous by design and programming-centric, engaging students with fundamentals of computer science by way of hands-on programming projects. The computational-thinking skills that students ultimately acquire are broadly applicable.

Among this course's objectives is to supply students with a comprehensive introduction to the fundamentals of the discipline of computer science. We will do so using programming in several different languages as a vehicle to introduce these fundamentals, including such topics as algorithms, abstraction, data, global impact, and internet technologies. Though the course is programming-heavy, it should be stressed that this is not a "programming course"; rather, this course should be considered one of problem-solving, creativity, and exploration. By year's end, students will have a richer understanding of the key principles of the discipline of computer science. They will be able to speak intelligently about how computers work and how they enable us to become better problem-solvers, and will hopefully be able to communicate that knowledge to others.

Whether students elect to take no other computer science courses in their lifetime or consider this class the first step in a longer course of study, it is our sincere hope that students feel more comfortable with—and indeed sometimes skeptical of—the technologies that surround us each day.

PREREQUISITES

The only background required for CS50 for AP Computer Science Principles is completion of Algebra I or its equivalent.

RECOMMENDED BOOKS

No books are required for this course. However, students may wish to supplement their preparation for or review of some material with self-assigned readings relevant to the material from either of the books below. The first is intended for those inexperienced in (or less comfortable with the idea of) programming. The second is intended for those experienced in (or more comfortable with the idea of) programming. Realize that free, if not superior, resources can be found on the course's website or on the internet more generally.

For Those Less Comfortable

C Programming Absolute Beginner's Guide, 3rd Edition
Greg Perry, Dean Miller

Pearson Education, 2014
ISBN 0-789-75198-4

For Those More Comfortable
Programming in C, 4th Edition
Stephen G. Kochan
Pearson Education, 2015
ISBN 0-321-77641-0

The following book is recommended for those interested in understanding more about how their own computers work, for personal edification.

How Computers Work, 10th Edition
Ron White
Que Publishing, 2014
ISBN 0-7897-4984-X

Lastly, the following book is recommended for aspiring hackers—those interested in programming techniques and low-level optimization of code that goes beyond the scope of this course.

Hacker's Delight, 2nd Edition
Henry S. Warren, Jr.
Pearson Education, 2013
ISBN 0-321-84268-5

PROGRAMMING ENVIRONMENTS

Several programming languages are taught in the course, and students are able to program in all of them in an environment designed specifically for the course called CS50 IDE. Students will need to sign up for a (free) account on edX (edx.org) in order to use CS50 IDE.

CS50 IDE is a web-based utility (hosted on a platform known as Cloud9) with cloud storage, meaning students will be able to work on the course's programming exercises at home, school, or anywhere they have an internet connection. Instructions for setting up and using CS50 IDE are provided in the first assignment requiring its use.

Additionally, students will use a drag-and-drop programming language called *Scratch* for some of the course's early material. *Scratch* is similarly a web-based environment, and it can be accessed by visiting https://scratch.mit.edu/.

ACADEMIC HONESTY

This course's philosophy on academic honesty is best stated as "**be reasonable.**" The course recognizes that interactions with classmates and others can facilitate mastery of the course's material. However, there remains a line between enlisting the help of another and submitting the work of another. This policy characterizes both sides of that line.

The essence of all work that you submit to this course must be your own. Collaboration on problems is not permitted (unless explicitly stated otherwise) except to the extent that you may ask classmates and others for help so long as that help does not reduce to another doing your work for you. **Generally speaking, when asking for help, you may show your code or writing to others, but you may not view theirs, so long as you and they respect this policy's other constraints.**

Collaboration on the course's quizzes and tests is not permitted at all. Collaboration on the Create Performance Task is permitted to the extent prescribed by its specifications.

Below are examples that inexhaustibly characterize acts that the course considers *reasonable* and *not reasonable*. If in doubt as to whether some act is reasonable, do not commit it until you solicit and receive approval in writing from your instructor. If a violation of this policy is suspected and confirmed, your instructor reserves the right to impose an appropriate penalty.

Reasonable

- Communicating with classmates about problems in English (or some other spoken language).
- Discussing the course's material with others in order to understand it better.
- Helping a classmate identify a bug in his or her code, such as by viewing, compiling, or running his or her code, even on your own computer.
- Incorporating snippets of code that you find online or elsewhere into your own code, provided that those snippets are not themselves solutions to assigned problems and that you cite the snippets' origins (as via comments in your code).
- Reviewing past years' quizzes, tests, and solutions thereto.
- Sending or showing code that you've written to someone, possibly a classmate, so that he or she might help you identify and fix a bug.
- Sharing snippets of your own solutions to problems online so that others might help you identify and fix a bug or other issue.
- Turning to the web or elsewhere for instruction beyond the course's own, for references, and for solutions to technical difficulties, but not for outright solutions to problems or your own Create Task.
- Whiteboarding solutions to problems with others using diagrams or pseudocode but not actual code.
- Working with (and even paying) a tutor to help you with the course, provided the tutor does not do your work for you.

Not Reasonable

- Accessing a solution to some problem prior to (re-)submitting your own.
- Asking a classmate to see his or her solution to a problem before (re-)submitting your own.
- Decompiling, deobfuscating, or disassembling the sample solutions to problems available in CS50 IDE.
- Failing to cite (as with comments) the origins of code, writing, or techniques that you discover outside of the course's own lessons and integrate into your own work, even while respecting this policy's other constraints.
- Giving or showing to a classmate a solution to a problem when it is he or she, and not you, who is struggling to solve it.
- Looking at another individual's work during a quiz or test.

- Paying or offering to pay an individual for work that you may submit as (part of) your own.
- Providing or making available solutions to problems to individuals who might take this course in the future.
- Searching for, soliciting, or viewing a quiz's questions or answers prior to taking the quiz.
- Searching for or soliciting outright solutions to problems online or elsewhere.
- Splitting a problem's workload with another individual and combining your work (unless explicitly authorized by the problem itself).
- Submitting (after possibly modifying) the work of another individual beyond allowed snippets.
- Submitting the same or similar work to this course that you have submitted or will submit to another.
- Using resources during a quiz or test beyond those explicitly allowed in the quiz's or test's instructions.
- Viewing another's solution to a problem and basing your own solution on it.

ASSESSMENT

Because computer science is not a discipline that only lends itself to questions of *right* and *wrong* but also *how* and *why*, this course's assessment policy is designed to try to answer some or all of these questions. The course's problems are evaluated along the three axes of correctness, design, and style.

Correctness	To what extent is your submission consistent with our specification and free of bugs or errors?
Design	To what extent is your submission written well (i.e., clearly, efficiently, elegantly, and/or logically)?
Style	To what extent is your submission readable (i.e., code is commented and intended with aptly-named variables)?

To obtain a passing grade in the course, all students must ordinarily submit all assigned problems unless otherwise granted an exception in writing by the instructor.

OVERVIEW

Consistent with the AP Computer Science Principles curriculum framework, the course's material is organized around seven so-called "big ideas" as well as six computational thinking practices. The seven big ideas are:

1. Creativity
2. Abstraction
3. Data and Information
4. Algorithms
5. Programming
6. The Internet
7. Global Impact

And the six computational thinking practices are:

1. Connecting Computing
2. Creating Computational Artifacts
3. Abstracting
4. Analyzing Problems and Artifacts
5. Communicating (both orally and in writing)
6. Collaborating

CORE UNITS

Unit 0

(Completion Time: 3 weeks)

In this unit, students will learn about computer hardware and architecture, the language of computers – binary, how information is encoded so that humans can understand it, and begin to explore the ways in which computers process information.

Topics within this Unit

0	Computers and Computing	3	Logic and Processors
1	How Computers Work	4	Memory
2	Binary and ASCII	5	Algorithms

Learning Objectives Covered in these Topics

LO 2.1.1	LO 3.1.1	LO 7.2.1
LO 2.1.2	LO 4.1.2	
LO 2.2.3	LO 4.2.4	

Assignments

0 Around the House (Computers and Computing)

Students explore the devices in their home to find "non-traditional" computers. In no more than 400 words, they'll describe these devices in detail. They will answer questions such as:

- *What does the device look like?*
- *What kind of data does it accept?*
- *How does it process that data?*
- *What is the result of that processing?*

- LO 7.1.1 [P4] **[CR1a]** **[CR1d]** **[CR2g]**

1 Tech Spotlight (Memory)

Students research technological innovations and apply their newfound knowledge of computer hardware. In no more than 600 words, students will expound on this technology. Their objective is to provide the reader with a well-rounded, unbiased summary of this innovation and the abstractions used in its creation. In writing their response, students will consider:

- *What is this technology called?*
- *What does it do?*
- *How does someone use this technology?*

- *How is its quality of performance commonly measured? (e.g. in megabytes (MB), gigahertz (GHz), etc.)*
- *How does the recent news about the technology change the product or service?*
- *What older form of technology does it replace, if any?*
- *How has this technology impacted your life, for better or worse?*
- *How has this technology impacted society at large, for better or worse?*

- LO 1.2.5 [P4], LO 2.2.3[P3], LO 5.1.1 [P2], LO 5.2.1 [P3], LO 7.1.1 [P4], LO 7.2.1 [P1], LO 7.3.1 [P4], LO 7.4.1 [P1], LO 7.5.2 [P5] **[CR1a] [CR1d] [CR1e] [CR2a] [CR2g]**

2 Everyday Algorithms (Algorithms)

Students will write an algorithm in sentence form and in pseudocode for how to complete a task that they do on a daily basis such as brushing their teeth. Students should strive to accurately describe that algorithm without ambiguity in a spoken language.

- LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.2.1 [P3] **[CR1b] [CR1e] [CR2d]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2g] – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Performance Task 1: Explore – Impact of Computing Innovations

(Completion Time: 2 weeks)

Students will start the through-course assessment *Explore—Impact of Computing Innovations*. Students are allocated eight hours of in-class time for this exercise. **[CR3]**

[CR3] – Students are provided the required amount of class time to complete the AP Through-Course Assessment *Explore – Impact of Computing Innovations* Performance Task.

Unit 1

(Completion Time: 4 weeks)

In this unit, students will learn the fundamentals of computer programming, to permit them to begin to manipulate information and data and command a computer to do calculations they wish for it to perform.

Topics within this Unit

0 Pseudocode
1 Scratch
2 Syntax
3 Variables

4 Data Types
5 Operators
6 Boolean Expressions and Conditionals
7 Loops

Learning Objectives Covered in these Topics

LO 1.2.1	LO 2.1.1	LO 4.2.4
LO 1.2.2	LO 2.2.2	LO 5.1.1
LO 1.2.3	LO 2.2.3	LO 5.2.1
LO 1.3.1	LO 4.1.2	LO 5.5.1

Assignments

0 Scratch (Scratch)

Students will use the drag-and-drop programming language called Scratch to implement a project of their choice (be it an animation, a game, interactive art, or anything else), subject only to the following requirements:

- *have at least two sprites, at least one of which must resemble something other than a cat*
- *have at least three scripts total (i.e., not necessarily three per sprite)*
- *use at least one condition, one loop, and one variable*
- *use at least one sound*

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.3.1 [P2], LO 2.1.2 [P5], LO 2.2.2 [P3], LO 4.1.1 [P2], LO 5.1.1 [P2], LO 5.5.1 [P1] **[CR1b] [CR1c] [CR1e] [CR2a] [CR2b] [CR2d] [CR2e]**

1 Hello (Syntax)

Students learn the syntax specific to C. Here, they write their first program in a web-based programming environment called the CS50 IDE.

- LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2] **[CR1a] [CR1b] [CR2b] [CR2d] [CR2e]**

2 Fahrenheit (Operators)

Students will write a program that converts a temperature in Celsius to Fahrenheit and explore bugs that might arise when dealing with imprecision relating to floats and division in C.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR2b] [CR2d] [CR2e]**

3 Cash (Boolean Expressions and Conditionals)

This activity introduces students to greedy algorithms. Here, they will write a program that first asks the user how much change is owed and then spits out the minimum number of coins with which said change can be made.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 2.2.2 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1[P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR2b] [CR2d] [CR2e]**

4 Pennies (Loops)

Students will create a file that calculates the amount that the user will have received in total by the end of the month (not just on the last day) if some initial amount is doubled on every day but the first, expressed not as pennies but as dollars and cents.

- LO 2.1.1 [P3], LO 4.1.1 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR2b] [CR2d] [CR2e]**

5 ISBN (Loops)

Students will further build on the abstractions available to them in C. Here, they will explore iteration and loops by writing a program that prompts the user for an ISBN-10 and then reports (via printf) whether the number's legit. The program's last line of output should be either yes or no, nothing more, nothing less.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2b] [CR2d] [CR2e]**

6 Mario (Loops)

Students further their understanding of loops and their familiarity with the syntax of C, by creating a program that outputs the famous Mario pyramid using spaces and hashes.

- LO 1.1.1 [P2], LO 1.2.5 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

7 Credit (Loops)

Students will put the concepts from Unit 1 together, using loops, iteration, booleans, and data types to implement a program that prompts the user for a credit card number and then reports (via printf) whether it is a valid American Express, MasterCard, or Visa card number, per the definitions of each's format.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2b] – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Unit 2:

(Completion Time: 5 weeks)

In this unit, students will expand upon their knowledge of the fundamentals of computer programming, and begin building abstractions of their own, understanding how to represent collections of data and write functions/methods/subroutines to allow their code to be more portable.

Topics within this Unit

0 Compiling

1 Functions

2	Arrays and Strings	5	Libraries
3	Command-Line Interactions	6	Typecasting
4	Exit Codes	7	Bugs and Debugging

Learning Objectives Covered in these Topics

LO 1.2.5	LO 2.2.3	LO 5.2.1
LO 2.1.1	LO 3.3.1	LO 5.3.1
LO 2.2.1	LO 4.2.4	LO 5.4.1
LO 2.2.2	LO 5.1.2	LO 5.5.1

Assignments

0 Me, Myself, and UI (Compiling)

Students will write a short essay to compare and contrast two interfaces and the different ways users have interacted with machines, then they will hypothesize on the future of human-machine interaction.

- LO 7.1.1 [P4], LO 7.2.1 [P1], LO 7.3.1 [P4], LO 7.4.1 [P1], LO 7.5.2 [P5] **[CR1a] [CR1c] [CR1d] [CR2b] [CR2g]**

1 Reverse Engineer (Compiling)

Students learn about the abstractions between machine code (0s and 1s), assembly code, source code, and the other abstractions between human and machine. Students will have to reverse engineer the C code that will result in the assembly code that we provide them with.

- LO 2.1.1 [P3], LO 2.1.2 [P5], LO 2.2.3 [P3], LO 5.3.1 [P3] **[CR1c] [CR1e] [CR2b]**

2 Initials (Arrays and Strings)

Students put their knowledge of arrays, string manipulation, and loops to the test, by creating a program that outputs the initials of the name that was entered.

- LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1c] [CR1d] [CR2b] [CR2d] [CR2e]**

3 Old Friends (Command-Line Interaction)

Students begin to interact with their programs at the command-line, allowing them to run differently each time, instead of always doing the same thing. Here, students modify some of the previous problems to allow them to be run from the command line.

- LO 1.2.2 [P2], LO 1.2.3 [P2], LO 5.1.1 [P2], LO 5.1.2 [P2] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR2a] [CR2b] [CR2d] [CR2e]**

4 Calc (Command-Line Interaction)

Students continue to interact with their programs at the command-line. They will implement the basic features of calculator including addition, subtraction, multiplication, division, and modulo.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 2.1.2 [P5], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

5 Caesar (Command-Line Interaction)

Students dive into cryptography—the transformation of "plaintexts" to instead be secret messages, and how we can use machines to do this for us. Here they implement their own version of a caesar cipher.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 2.1.1 [P3], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

6 Vigenére (Bugs and Debugging)

Furthering their understanding of cryptography, students will create a cipher more secure than a caesar cipher, a vigenere cipher, where a keyword is used to encrypt the message.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 2.1.1 [P3], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

7 Crack (Bugs and Debugging)

After learning about encrypting "plaintext," students will explore the opposite. They will create a program, using varying levels of abstraction such as functions and libraries to help them decrypt encrypted passwords.

- LO 1.1.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 2.1.1 [P3], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2d] [CR2e]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2b] – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2g] – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Unit 3

(Completion Time: 4 weeks)

In this unit, students begin to consider different algorithms and *how* they process information, appreciating the differences between algorithms and gaining the vocabulary necessary to speak on their relative merits and disadvantages. They will also be introduced to the concept of undecidability, the fact that computers are not capable of answering every question we ask them.

Topics within this Unit

0 Linear Search
1 Bubble Sort
2 Selection Sort
3 Insertion Sort
4 Binary Search
5 Computational Complexity

6 Unsolvable Problems

7 Models and Simulations

Learning Objectives Covered in these Topics

LO 2.3.2	LO 4.2.2	LO 4.2.4
LO 4.2.1	LO 4.2.3	LO 5.2.1

Assignments

0 Find (Selection Sort)

Students will implement a random number generator and students will be introduced to header files and implement a searching algorithm to best find a number in a randomly generated list of numbers.

- LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.2.5 [P4], LO 3.1.1 [P4], LO 3.2.1 [P1], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 4.2.1 [P1], LO 4.2.4 [P4], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.2.1 [P3], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

1 Music (Selection Sort)

Students take their newfound knowledge of functions and organizing data, and aided by some distribution code that implements the basic framework for them, to collaborate on implementing the classic Game of Fifteen with user-interactivity, while explaining their implementations with other groups.

- LO 1.1.1 [P2], LO 1.2.3 [P2], LO 1.2.4 [P6], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.2 [P2], LO 5.1.3 [P6], LO 5.2.1 [P3], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

2 Game of Fifteen (Selection Sort)

Students take their newfound knowledge of functions and organizing data, and aided by some distribution code that implements the basic framework for them, to collaborate on implementing the classic Game of Fifteen with user-interactivity, while explaining their implementations with other groups.

- LO 1.1.1 [P2], LO 1.2.3 [P2], LO 1.2.4 [P6], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.2 [P2], LO 5.1.3 [P6], LO 5.2.1 [P3], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

3 Sort Race (Insertion Sort)

Having tackled the various sorting algorithms, students will collaborate to create a program to run simulations of three different sorting algorithms to see which algorithms run better under different starting conditions and sizes, then explain these variations.

- LO 1.1.1 [P2], LO 1.2.3 [P2], LO 1.2.4 [P6], LO 1.2.5 [P4], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.1.1 [P4], LO 3.1.2 [P6], LO 3.1.3 [P5], LO 3.2.2 [P3], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 4.2.1 [P1], LO 4.2.4 [P4], LO 5.1.2 [P2], LO 5.1.3 [P6], LO 5.2.1 [P3], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2c] [CR2d] [CR2e]**

4 Analyze This (Insertion Sort)

Students take an opportunity to reflect on their growth through the course so far, asking themselves which concepts they previously struggled with and what challenges they further anticipate, while also acknowledging their tremendous growth. In 500-1,000 words, students will discuss their growth with peers, then individually prepare an essay on your experience, expectations, growths, triumphs, and struggles up through this point in the course.

- LO 3.1.2 [P6] **[CR1d] [CR1f] [CR2c] [CR2g]**

5 Scramble (Part 1) (Computational Complexity)

Scramble is a game where users must find as many words as they can in a grid of letters; the only rule being that each successive letter must be adjacent to its predecessor. Here, students will implement the basic logic for this game.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 2.2.1[P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

6 Simulate! (Simulation)

In this writing problem, students research a computer simulation of their choice. They will explain how the simulation and the benefits and disadvantages of using it. Does the program account for all the features it is trying to model? Does the model rely on any assumptions? Are there downsides to using a program instead of testing in the real-world? What are those downsides?

- LO 2.3.1 [P3], LO 2.3.2 [P3], LO 7.2.1 [P1], LO 7.3.1[P4] **[CR1a] [CR1d] [CR2a] [CR2g]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR1f] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2b] – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2c] – Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2g] – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Unit 4

(Completion Time: 4 weeks)

As students begin to wrap up their introduction to C (except for those in the Fast-Track Units, below), they are challenged to consider questions as to what makes a solution good, and what makes a program beautiful. In this unit in particular, we challenge students to consider not just right and wrong, but how and why, discussing design trade-offs and why sometimes indeed the discipline of computer science boils down to determining what trade-off is appropriate under a given set of circumstances.

Topics within this Unit

0 Principles of Good Design
1 ncurses
2 Structures and Encapsulation
3 Recursion
4 Merge Sort
5 Hexadecimal
6 File I/O
7 Images
8 Version Control and Collaboration

Learning Objectives Covered in these Topics

LO 1.1.1	LO 2.1.1	LO 3.1.2
LO 1.2.2	LO 2.1.2	LO 3.3.1
LO 1.2.3	LO 2.2.1	LO 4.2.4
LO 1.2.4	LO 2.2.2	LO 5.1.2
LO 1.2.5	LO 2.2.3	LO 5.1.3
LO 1.3.1	LO 3.1.1	LO 5.3.1

Assignments

0 Sudoku (Structures and Encapsulation)

Students use their new found knowledge of graphics and structures to implement the logic behind sudoku in addition to some supplementary features that they choose.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.3 [P2], LO 1.2.5 [P4], LO 2.1.1 [P3], LO 5.1.1 [P2], LO 5.1.2 [P2] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR2a] [CR2b] [CR2d] [CR2e]**

1 Finder (File I/O)

Students learn about their file system and how they can navigate it programmatically, searching through their own directories and further explore what is going on underneath the hood of their own searches and the abstractions that they are exposed to on their machines. Students will implement their own search function to iterate through directories to find specific strings of characters.

- LO 1.3.1 [P2], LO 2.1.1 [P3], LO 2.1.2 [P5], LO 2.2.3 [P3], LO 3.2.2 [P3], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR2a] [CR2b] [CR2c] [CR2e]**

3 Whodunit (Images)

Students explore images in depth and the varying levels of abstraction used to represent an image, rooting back to the individual bits that compose the pixels within an image. They will both individually and in teams, modify bitmap images to extract a hidden image. Additionally, they will answer some questions about images more generally.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 2.1.1 [P3], LO 2.1.2 [P5], LO 3.1.1 [P4], LO 3.1.3 [P5], LO 3.2.1 [P1], LO 4.1.1 [P2], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

4 Resize (Images)

Diving deeper into bitmap manipulation, students will create a program that takes in a 24-bit uncompressed BMPs and scales it larger by a factor of n.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 2.1.1 [P3], LO 2.1.2 [P5], LO 3.1.1 [P4], LO 3.1.3 [P5], LO 3.2.1 [P1], LO 4.1.1 [P2], LO 5.1.2 [P2], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2d] [CR2e]**

6 Recover (Images)

In this problem, students will receive the file of a corrupted memory card storing 50 jpegs. They will work in groups to use their knowledge of file I/O to read the images from the memory card and write them to new files, thus restoring the lost images.

- LO 1.2.2 [P2], LO 1.2.3 [P2], LO 1.2.4 [P6], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.1.2 [P6], LO 3.2.1 [P1], LO 4.1.1 [P2], LO 4.2.4 [P4], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.1.3 [P6], LO 5.3.1 [P3], LO 5.4.1 [P4], LO 5.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2c] [CR2d] [CR2e]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR1f] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2b] – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2c] – Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Unit 5

(Completion Time: 3 weeks)

At this point in the course, we transition from programming in a mostly command-line environment to taking our applications to scale via the Internet. First, however, students are introduced to the technologies underpinning this thing we know as "the Internet" before beginning to explore web programming by building simple pages of their own and making them accessible to the world via CS50 IDE.

Topics within this Unit

0	Internet Basics	5	HTTP
1	IP Addresses	6	Trust Models
2	DNS and DHCP	7	Cybersecurity
3	Routers	8	HTML
4	TCP and IP	9	CSS

Learning Objectives Covered in these Topics

LO 1.2.5	LO 6.1.1	LO 7.3.1
LO 2.2.2	LO 6.2.1	LO 7.4.1
LO 3.2.1	LO 6.2.2	
LO 3.3.1	LO 6.3.1	

Assignments

0 Be the Teacher (HTTP)

The technologies of the internet can be complex, so students are challenged to explain in writing things concisely to a lay audience, cementing their understanding of these technologies by having to discuss them more casually. In 1500 words students will explain how the internet works to 3rd graders.

- LO 6.1.1 [P3], LO 6.2.1 [P5], LO 6.2.2 [P4], LO 7.3.1 [P4] **[CR1c]** **[CR1d]** **[CR1e]** **[CR2f]**

1 Defender of the Web (Cybersecurity)

Students explore the notions of cyberattacks and cybersecurity, and investigate in more detail some of the common types of attacks that impact websites today. In 750-1,000 words, students should cover the following:

- *What is the name of the attack? What type of attack is it?*
- *Where did it come from? Who created it (if known), and why?*
- *How did we find out about it - how was it caught?*
- *What types of companies or individuals does it target?*
- *How does it work? What components of the network does it attack, and from which end (client or server)?*
- *What damage is it capable of doing? What information does it target?*
- *Has a fix been found? How does it work? Has it been implemented in all websites/servers with potential vulnerabilities?*
- *If applicable, how can we defend ourselves against this attack?*

- LO 6.1.1 [P3], LO 6.2.1 [P5], LO 6.2.2 [P4], LO 6.3.1 [P1], LO 7.1.1 [P4], LO 7.2.1 [P1], LO 7.3.1 [P4], LO 7.4.1 [P1] **[CR1a]** **[CR1c]** **[CR1d]** **[CR1e]** **[CR2f]** **[CR2g]**

2 </Unit 5> (CSS)

Students create their own web pages, learn about permissions schemes, and make their creations accessible to the world.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 1.3.1 [P2], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.3.1 [P3], LO 6.1.1 [P3] **[CR1b]** **[CR1c]** **[CR2a]** **[CR2e]** **[CR2f]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR1f] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

[**CR2f**] – Students are provided with opportunities to meet learning objectives within Big Idea 6: The Internet. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[**CR2g**] – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Unit 6

(Completion Time: 5 weeks)

Lastly, students build upon their knowledge gained in the course to learn several new programming languages with abstractions built in that allow them to go far beyond what simply C and Scratch are able to do. They solve more complex problems that require processing large amounts of data and dealing with processes that scale, and see how these techniques can be applied to confront the challenges computer scientists will be contending with in the future.

Topics within this Unit

0 Python
1 Python for Web Programming
2 SQL
3 MVC
4 JavaScript
5 Ajax
6 Artificial Intelligence and Machine Learning
7 Virtual and Augmented Reality

Learning Objectives Covered in these Topics

LO 2.3.1	LO 3.1.1	LO 6.2.2
LO 3.1.3	LO 6.1.1	LO 7.1.1
LO 3.2.1	LO 6.2.1	LO 7.5.1

Assignments

0 Sentimental (Python)

Students will re-implement Mario, Cash, and Caesar in Python to gain familiarity with Python syntax and the affordances that higher level programming languages offer.

- LO 1.2.2 [P2], LO 1.2.3 [P2], LO 1.2.5 [P4], LO 3.1.3 [P5], LO 5.1.1 [P2], LO 5.1.2 [P2], LO 5.3.1 [P3], LO 5.4.1 [P4]

1 Similarities (Python for Web Programming)

Students will write a program to determine segments of similar code between two sample submissions.

- LO 1.2.2 [P2], LO 1.2.5 [P4], LO 2.2.1 [P2], LO 2.2.2 [P3], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.5.1 [P1]

2 C$50 Finance (SQL)

Students will work in groups to construct their own stock-trading website (pulling real stock prices from Yahoo! finance), working with databases and managing user information securely.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.2.3 [P2], LO 1.2.4 [P6], LO 1.3.1 [P2], LO 2.1.1 [P3], LO 2.2.3 [P3], LO 3.1.2 [P6], LO 3.1.3 [P5], LO 3.2.1 [P1], LO 3.3.1 [P4], LO 4.1.1 [P2], LO 4.1.2 [P5], LO 5.1.1 [P2], LO 5.1.3 [P6], LO 5.3.1 [P3], LO 5.5.1 [P1], LO 6.3.1 [P1], LO 7.5.1 [P1] **[CR1a]** **[CR1b]** **[CR1c]** **[CR1d]** **[CR1e]** **[CR1f]** **[CR2a]** **[CR2b]** **[CR2c]** **[CR2d]** **[CR2e]** **[CR2f]** **[CR2g]**

3 Mashup (Ajax)

Standing on the shoulders of those who came before them, students will collaborate to build upon Google Developer tools to construct an interactive website for displaying news on a map.

- LO 1.1.1 [P2], LO 1.2.1 [P2], LO 1.2.2 [P2], LO 1.2.4 [P6], LO 1.3.1 [P2], LO 2.1.1 [P3], LO 2.2.2 [P3], LO 2.2.3 [P3], LO 3.1.2 [P6], LO 3.1.3 [P5], LO 3.2.1 [P1], LO 5.1.1 [P2], LO 5.1.3 [P6], LO 5.3.1 [P3], LO 5.5.1 [P1], LO 7.5.1 [P1] **[CR1a] [CR1b] [CR1c] [CR1d] [CR1e] [CR1f] [CR2a] [CR2b] [CR2c] [CR2e] [CR2g]**

[CR1a] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
[CR1b] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
[CR1c] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
[CR1d] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
[CR1e] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
[CR1f] – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
[CR2a] – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2b] – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2c] – Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2d] – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2e] – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2f] – Students are provided with opportunities to meet learning objectives within Big Idea 6: The Internet. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
[CR2g] – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

Performance Task 2: Create – Applications from Ideas

(Completion Time: 3 weeks)

After competing Unit 6, students complete the through-course assessment *Create—Applications from Ideas*. Students are allocated twelve hours of in-class time for this exercise. **[CR4]**

[CR4] – Students are provided the required amount of class time to complete the AP Through-Course Assessment *Create – Applications from Ideas* Performance Task.

FAST-TRACK UNITS

For students who find themselves acclimating well to the course's material, have prior background in computer science, or consider themselves among those more comfortable, the course provides a fast-track that allows these students to explore material related to that covered in the course's core units.

The material in these units goes beyond the scope of the AP Computer Science Principles curriculum framework, but provides a venue for students to gain a deeper appreciation for the computational power of lower-level programming languages and an opportunity to explore data structures and data management techniques.

These fast-track units are designed to come, in the course's narrative, between Units 4 and 5. Because these units are strictly optional, no completion timeline is provided and no mapping to the AP Computer Science Principles curriculum framework is given.

Unit A

In this fast-track unit, students learn about different ways of managing sets of data, both for sets that will rapidly and boundlessly grow, and also for easier subsequent indexing and searching.

Visit https://ap.cs50.net/curriculum/A/ to view content and descriptions of the following:

Topics within this Unit

0 Stacks
1 Queues
2 Pointers
3 Dynamic Memory
4 Valgrind
5 Trees
6 Tries
7 Linked Lists
8 Hash Tables

Assignments

0 Calc 2.0 (Queues)
1 Leaky (Valgrind)
2 Autocomplete (Tries)
3 Decisions (Linked Lists)
4 Misspellings (Hash Tables)

Unit B

In this fast-track unit, students will complete their introduction to C by diving into some of its most complex syntax, and implement strategies for data compression, abstractions, API development, and interfacing for web-traffic using C.

Visit https://ap.cs50.net/curriculum/B/ to view content and descriptions of the following:

Topics within this Unit

0 Abstraction and API
1 Data Compression
2 Huffman Coding
3 Scalability
4 Collaboration

Assignments

0 Blockdude (Data Compression)
1 Puff (Huffman Coding)
2 Huff (Huffman Coding)
3 Degrees of Scalability (Degrees of Scalability)
4 Server (Server)