CS50 for AP Computer Science Principles

COURSE SYLLABUS – TABLE OF CONTENTS

CURRICULAR REQUIREMENTS

P1	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P1: Computational Solution Design.
P2	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P2: Algorithms and Program Development.
P3	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P3: Abstraction in Program Development.
P4	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P4: Code Analysis.
P5	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P5: Computing Innovations.
P6	The course provides opportunities for students to develop the skills related to Computational Thinking Practice P6: Responsible Computing.
CRD	The course provides opportunities to develop student understanding of Big Idea 1: Creative Development.
DAT	The course provides opportunities to develop student understanding of Big Idea 2: Data.
AAP	The course provides opportunities to develop student understanding of Big Idea 3: Algorithms and Programming.
CSN	The course provides opportunities to develop student understanding of Big Idea 4: Computing Systems and Networks.
IOC	The course provides opportunities to develop student understanding of Big Idea 5: Impact of Computing.
CI-1	The course provides an opportunity for students to investigate Computing Innovation 1 - Basic Home Computing.
CI-2	The course provides an opportunity for students to investigate Computing Innovation 2 - Large Scale Data Processing.
CI-3	The course provides an opportunity for students to investigate Computing Innovation 3 - Cybersecurity and the Internet.
TCA	Students are provided at least twelve (12) hours of dedicated class time to complete the AP Create 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) GitHub (github.com/join) account in order to use CS50 IDE.

CS50 IDE is a web-based utility (hosted on a platform known as AWS 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 two axes: correctness and style.

Correctness	To what extent is your submission consistent with our specification and free of bugs or errors?
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 five so-called "big ideas" as well as six computational thinking practices. The five big ideas are:

1. Creative Development (CRD)
2. Data (DAT)
3. Algorithms and Programming (AAP)
4. Computing Systems and Networks (CSN)
5. Impact of Computing (IOC)

And the six computational thinking practices are:

P1. Computational Solution Design
P2. Algorithms and Program Development
P3. Abstraction in Program Development
P4. Code Analysis
P5. Computing Innovations
P6. Responsible Computing

CS50 CORE CURRICULUM

Chapter 0

In this chapter, students will learn about how data is represented in their computer and 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 Chapter

0	Binary	3	Pseudocode
1	ASCII	4	Scratch
2	Algorithms		

Assignments

0 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) ***[P1] [P2] [P3] [P4] [P6]****, 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*

Big Ideas Covered in these Topics

CRD-1.A	CRD-2.C	CRD-2.E
CRD-2.A	CRD-2.D	CRD-2.J

DAT-1.A	AAP-1.A	AAP-2.C
DAT-1.B	AAP-2.A	AAP-2.F
DAT-1.C	AAP-2.B	AAP-3.A

Chapter 1

In this chapter, 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 Chapter

0	Syntax	3	Operators
1	Variables	4	Boolean Expressions and Conditionals
2	Data Types	5	Loops

Assignments

0 Hello

Students learn the syntax specific to C. Here, they write their first program in a web-based programming environment called the CS50 IDE. ***[P1] [P3] [P4]***

1 Fahrenheit

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. ***[P1] [P3] [P2] [P4]***

2 Cash

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 outputs the minimum number of coins with which said change can be made. ***[P1]*** **[P3] [P2] [P4]**

3 Pennies

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. ***[P1] [P3] [P2] [P4]***

4 ISBN

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 legitimate. The program's last line of output should be either yes or no, nothing more, nothing less. ***[P1] [P3] [P2] [P4]***

5 Mario

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. ***[P1] [P2] [P3] [P4]***

6 Credit

Students will put the concepts from Chapter 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. ***[P1] [P2] [P3] [P4]***

Big Ideas Covered in these Topics

CRD-2.E	AAP-2.B	AAP-2.M
CRD-2.J	AAP-2.C	AAP-3.A
DAT-1.A	AAP-2.F	AAP-3.B
AAP-2.A	AAP-2.H	AAP-3.C

Chapter 2

In this chapter, students will expand upon their knowledge of the fundamentals of computer programming and begin building abstractions of their own. They'll also learn about strategies for debugging their own programs.

Topics within this Chapter

0	Compiling	4	Exit Codes
1	Debugging	5	Searching
2	Arrays and Strings	6	Computational Complexity
3	Command-Line Interactions	7	Computational Models

Assignments

0 Old Friends

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. ***[P3] [P4]***

1 Calc

Students continue to interact with their programs at the command line. They will implement the basic features of calculators including addition, subtraction, multiplication, division, and modulo. ***[P1] [P2] [P3] [P4]***

2 Caesar

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. ***[P1] [P2] [P3] [P4]***

3 Vigenére

Furthering their understanding of cryptography, students will create a cipher more secure than a caesar cipher, a vigenére cipher, where a keyword is used to encrypt the message. ***[P1] [P2] [P3] [P4]***

4 Crack

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. ***[P1] [P2] [P3] [P4] [P6]***

Big Ideas Covered in these Topics

CRD-2.J	AAP-2.C	AAP-4.A
DAT-1.A	AAP-3.B	AAP-4.B
DAT-2.E	AAP-3.C	IOC-1.A
AAP-2.A	AAP-3.D	IOC-2.A

Chapter 3

In this chapter, students will uncover some of the concepts that go on under the hood when we use different data structures like strings and arrays. Then students explore ways to store data of various types in a struct.

Topics within this Chapter

0	Functions	3	Memory
1	Tools for Debugging	4	Pointers
2	More on Strings	5	Structs

Assignments

0 Fifteen

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. ***[P1] [P2] [P3] [P4]***

1 Whodunit

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. ***[P1] [P2] [P3] [P4]***

2 Resize

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. ***[P1] [P2] [P3] [P4] [P5]***

3 Recover

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. ***[P1] [P2] [P3] [P4] [P6]***

Big Ideas Covered in these Topics

CRD-2.A	DAT-2.A	AAP-3.A
CRD-2.E	DAT-2.D	AAP-3.B
CRD-2.J	DAT-2.E	IOC-1.A
DAT-1.B	AAP-2.A	IOC-2.B

Chapter 4

As students begin to wrap up their time in C, they are challenged to consider more complex data structures. Students dive into the various pros and cons of the various data structures and which are better to use in various scenarios.

Topics within this Chapter

0	Valgrind	3	Hash Tables
1	More on Structs	4	Trees
2	Linked Lists	5	Tries

Assignments

0 Speller

Students use their new-found knowledge of data structures to implement a spell checker in C. They implement several functions that work within staff-provided distribution code and test their code with various text files. ***[P1] [P2] [P3] [P4]***

Bid Ideas Covered in these Topics

CRD-1.A	CRD-2.J	AAP-2.L
CRD-1.C	DAT-1.A	AAP-2.O
CRD-2.E	DAT-2.D	AAP-3.B
CRD-2.A	DAT-2.E	AAP-3.C
CRD-2.E	AAP-1.D	AAP-3.D

Chapter 5

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 Chapter

0	Internet Basics	3	Forms
1	HTTP	4	CSS
2	HTML	5	JavaScript

Assignments

0 Be the Teacher

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. ***[P3] [P5] [P6]***

1 Defender of the Web

Students explore the notions of cyberattacks and cybersecurity and investigate in more detail some of the common types of attacks that impact websites today. ***[P1] [P3] [P5] [P6]***

In accordance with ***Computing Innovation 3 [CI-3], students will explore Cybersecurity and the Internet and****:*

- *[A] Students will explain beneficial and harmful effects of at least one computing innovation on society, economy, or culture.*
- *[B] Students will identify the data used in at least one computing innovation and explain how the data is consumed, produced, or transformed by the given computing innovation.*
- *[C] Students will identify data privacy, security, or storage concerns for at least one computing innovation.*

Students should likewise cover the following (in a total of 750-1,000 words):

- *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?*

2 Homepage

Students create their own web pages, learn about permissions schemes, and make their creations accessible to the world. ***[P1] [P4] [P6]***

Big Ideas Covered in these Topics

CRD-1.A	AAP-3.D	IOC-1.A
CRD-2.A	CSN-1.B	IOC-1.B
DAT-2.E	CSN-1.C	IOC-1.F
AAP-3.C	CSN-1.D	IOC-2.B

Chapter 6

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 Chapter

0	Python	3	Loops
1	Conditionals	4	Data Types
2	Booleans	5	Functions

Assignments

0 Analyze This

Students will reflect on their experiences in the course. In 500-1000 words, they will talk about some of the challenges they encountered in the course and how they persevered through the problem.

1 Sentimental

Students will re-implement a subset of Hello, Mario, Cash, Credit, Caesar, Vigenére, and Crack in Python to gain familiarity with Python syntax and the affordances that higher level programming languages offer. ***[P1] [P3] [P4]***

2 Bleep

Students will implement a program that censors out a list of banned words by replacing them with asterixis. Here students gain familiarity with Python specific functionality, particularly around string manipulation. ***[P1] [P3] [P4] [P6]***

Big Ideas Covered in these Topics

CRD-2.D	DAT-2.A	IOC-1.A
CRD-2.F	DAT-2.D	IOC-1.E
CRD-2.H	DAT-2.E	IOC-1.F
DAT-1.A	AAP-1.C	AAP-3.D

Chapter 7

Students build upon their knowledge of web programming and Python to create web-based applications. They learn about structures for organizing their files for web applications, like MVC.

Topics within this Chapter

0	Flask	3	Events
1	MVC	4	Ajax
2	Autocomplete	5	JSON

Assignments

0 Similarities

Students will write a program to determine segments of similar code between two sample submissions. ***[P1] [P2] [P3] [P4] [P6]***

1 Survey

Students will implement a web application similar to Google Forms, whereby students implement a form that collects information and saves the user data to a CSV and displays the CSV to the user on a web page. ***[P1] [P2] [P3] [P4] [P5]***

Big Ideas Covered in these Topics

CRD-2.A	DAT-2.D	AAP-3.D
CRD-2.E	DAT-2.E	CSN-1.C
CRD-2.H	AAP-2.A	IOC-2.A
CRD-2.J	AAP-3.B	IOC-2.B

Chapter 8

Students learn about how web applications store data in databases. They solve 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 Chapter

0	Cookies	3	SQL
1	Sessions	4	Race Conditions
2	Databases		

Assignments

0 C$50 Finance (SQL)

Students will work in groups to construct their own stock-trading website (pulling real stock prices from a finance API), working with databases and managing user information securely. ***[P1] [P2] [P3] [P4] [P5] [P6]***

Big Ideas Covered in these Topics

CRD-1.C	DAT-2.D	AAP-2.M
CRD-2.E	DAT-2.E	AAP-3.C
CRD-2.F	AAP-2.A	AAP-3.D
CRD-2.H	AAP-2.B	IOC-1.F

AP MODULES

The material in these chapters include topics that are not covered in CS50 on campus but are essential to the AP Computer Science Principles Course.

These modules are less programming focused and can be integrated at any point in the curriculum.

Understanding Technology

In this module, students learn about how it all works underneath the hood and how to solve problems when something goes wrong, this course fills in the gaps, empowering students to use and troubleshoot technology more effectively.

Topics within this Module

0	Hardware	2	Multimedia
1	The Internet	3	Security

Assignments

0 Around the House

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. ***[P5] [P6]***

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?*

1 Tech Spotlight

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. ***[P1] [P3] [P5] [P6]***

In accordance with ***Computing Innovation 1[CI-1], students will explore Technology around the Home and****:*

- *[B] Students will Identify the data used in at least one computing innovation and explain how the data is consumed, produced, or transformed by the given computing innovation.*

In addition, 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?*

2 Everyday 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. ***[P2] [P3]***

Big Ideas Covered in these Topics

CRD-2.J	CSN-1.A	IOC-1.A
DAT-1.A	CSN-1.B	IOC-1.F
DAT-1.B	CSN-1.C	IOC-2.A
DAT-2.E	CSN-1.D	IOC-2.B

Data Science

In this module, students learn about big data. Students learn about how to collect and analyze data responsibly and how human bias can affect computation artifacts.

Topics within this Module

0 Collecting Data

[P1] [P4] [P5]

1 Analyzing Data

[P1] [P4] [P5]

Big Ideas Covered in these Topics

CRD-2.H	DAT-2.A	IOC-1.A
DAT-1.A	DAT-2.D	IOC-1.D
DAT-1.B	DAT-2.E	IOC-1.F

Impact of Computing

In this module, students learn about how computing affects society. Students also dive into the development process from a high level and the ethics and legalities around computer science.

Topics within this Module

0 The Development Process
1 Scaling
2 Models
3 Simulations
4 The Digital Divide
5 Ethics and Legalities of Computing

Assignments

0 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? ***[P1] [P5] [P6]***

1 Degrees of Scalability

In this writing problem, students show how scalable real-world applications are. ***[P1] [P5] [P6]***

In accordance with ***Computing Innovation 2 [CI-2], students will explore Large-Scale Data Processing and****:*

- *[A] Students will explain beneficial and harmful effects of at least one computing innovation on society, economy, or culture.*
- *[C] Students will identify data privacy, security, or storage concerns for at least one computing innovation.*

Additional questions include but are not limited to:

- *How do companies handle big data and large amounts of users?*
- *What factors affect the scalability of a product?*

Big Ideas Covered in these Topics

CRD-2.A	CSN-1.B	IOC-1.D
DAT-1.A	IOC-1.A	IOC-1.E
DAT-2.E	IOC-1.B	IOC-1.F
AAP-3.F	IOC-1.C	IOC-2.B

Create Performance Task

After completing Chapter 8, students should be provided at least twelve (12) hours of dedicated class time to complete the Create: Applications from Ideas Performance Task [**TCA**].

Recommended Order

1. Understanding Technology Module
2. Chapter 0
3. Chapter 1
4. Chapter 2
5. Chapter 3
6. Chapter 4
7. Data Science Module
8. Chapter 5
9. Chapter 6
10. Chapter 7
11. Chapter 8
12. Create Performance Task
13. Impact of Computing Module