Randolph Township Schools
Randolph High School

AP Computer Science Principles

"Computer Science is a science of abstraction -creating the right model for a problem and devising the appropriate mechanizable techniques to solve it."

- A. Aho and J. Ullman

Department of
Science, Technology, Engineering and Math
Stacy Winters, Supervisor

Curriculum Committee
Sally Snelson
Michael Pignaloso III

Curriculum Developed:
July 2016

Date of Board Approval:
September 13, 2016

Randolph Township Schools
Department of Science, Technology, Engineering and Mathematics
AP Computer Science Principles

Table of Contents

Randolph Township Schools

Mission Statement

We commit to inspiring and empowering all students in Randolph schools to reach their full potential as unique, responsible and educated members of a global society.

Randolph Township Schools
Affirmative Action Statement

Equality and Equity in Curriculum

The Randolph Township School district ensures that the district's curriculum and instruction are aligned to the state's standards. The curriculum provides equity in instruction, educational programs and provides all students the opportunity to interact positively with others regardless of race, creed, color, national origin, ancestry, age, marital status, affectional or sexual orientation, gender, religion, disability or socioeconomic status.

N.J.A.C. 6A:7-1.7(b): Section 504, Rehabilitation Act of 1973; N.J.S.A. 10:5; Title IX, Education Amendments of 1972

RANDOLPH TOWNSHIP BOARD OF EDUCATION
EDUCATIONAL GOALS
VALUES IN EDUCATION

The statements represent the beliefs and values regarding our educational system. Education is the key to self-actualization, which is realized through achievement and self-respect. We believe our entire system must not only represent these values, but also demonstrate them in all that we do as a school system.

We believe:

- The needs of the child come first
- Mutual respect and trust are the cornerstones of a learning community
- The learning community consists of students, educators, parents, administrators, educational support personnel, the community and Board of Education members
- A successful learning community communicates honestly and openly in a non-threatening environment
- Members of our learning community have different needs at different times. There is openness to the challenge of meeting those needs in professional and supportive ways
- Assessment of professionals (i.e., educators, administrators and educational support personnel) is a dynamic process that requires review and revision based on evolving research, practices and experiences
- Development of desired capabilities comes in stages and is achieved through hard work, reflection and ongoing growth

Randolph Township Schools

AP Computer Science Principles

Introduction

The AP Computer Science Principles course is a full year course designed to be equivalent to a first-semester introductory college computing course. In this course, students will develop computational thinking skills vital for success across all disciplines, such as using computational tools to analyze and study data and working with large data sets to analyze, visualize, and draw conclusions from trends. The course is unique in its focus on fostering student creativity. Students are encouraged to apply creative processes when developing computational artifacts and to think creatively while using computer software and other technology to explore questions that interest them. They will also develop effective communication and collaboration skills, working individually and collaboratively to solve problems, and discussing and writing about the importance of these problems and the impacts to their community, society, and the world.

RANDOLPH TOWNSHIP SCHOOL DISTRICT
Curriculum Pacing Chart
AP Computer Science Principles

SUGGESTED TIME ALLOTMENT	UNIT NUMBER	CONTENT - UNIT OF STUDY
5 weeks	I	The Internet
6 weeks	II	Digital Information
6 weeks	III	Algorithms and Programming
4 weeks	IV	Big Data and Privacy
7 weeks	V	Building Apps
5 weeks	VI	Performance Tasks And AP Test Review
3 weeks	VII	Post AP Exam Projects/Programming

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT I: The Internet

<table>
<tr>
<td rowspan="7"><b>STANDARDS / GOALS:</b>

<b>CSTA K-12 Computer Science Standards</b>

CI.L3B:2 - Analyze the beneficial and harmful effects of computing innovations.

CI.L3B:4 - Summarize how computation has revolutionized the way people build real and virtual organizations and infrastructures.

CL.L2:3 - Collaborate with peers, experts and others using collaborative practices such as pair programming, working in project teams and participating in-group active learning activities.

CL.L2:4 - Exhibit dispositions necessary for collaboration: providing useful feedback, integrating feedback, understanding and accepting multiple perspectives, socialization.</td>
<td><b>ENDURING UNDERSTANDINGS</b></td>
<td><b>ESSENTIAL QUESTIONS</b></td>
</tr>
<tr>
<td>There are trade-offs when representing information as digital data.</td>
<td>• Is binary the best way to represent digital information?</td>
</tr>
<tr>
<td>Characteristics of the Internet influence the systems built on it.</td>
<td>• Are the ways data is represented and transmitted with computers laws of nature or a laws of man?</td>
</tr>
<tr>
<td>A variety of abstractions built upon binary sequences can be used to represent all digital data on the internet.</td>
<td>• Who and what is “in charge” of the Internet and how it functions?</td>
</tr>
<tr>
<td>Computing has a global affect -- both beneficial and harmful -- on people and society.</td>
<td>• In what ways are you or your family directly affected by computers and/or computer science?</td>
</tr>
<tr>
<td><b>KNOWLEDGE</b></td>
<td><b>SKILLS</b></td>
</tr>
<tr>
<td><b>Students will know:</b>
Computing innovations have impacted innovations in other fields and help to analyze the beneficial and harmful effects of computing.

There are a variety of abstractions used to represent data.</td>
<td><b>Students will be able to:</b>
Describe positive and negative effects of computing innovations.

Communicate with classmates about computing innovations in their lives.

Analyze the possibilities and limitations that arise when sending binary messages.</td>
</tr>
</table>

CT - Computational Thinking CT.L2:7 - Represent data in a variety of ways including text, sounds, pictures and numbers. CT.L2:8 - Use visual representations of problem states, structures and data (e.g., graphs, charts, network diagrams, flowcharts). CT.L2:9 - Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. CT.L2:14 - Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions. **Computer Science Principles** 2.1 - A variety of abstractions built upon binary sequences can be used to represent all digital data. 2.3 - Models and simulations use abstraction to generate new understanding and knowledge. 3.1 - People use computer programs to process information to gain insight and knowledge.	Data representation, storage, security, and transmission of data involve computational manipulation of information. The internet has a set of specific characteristics which are essential to the systems built on it. The abstractions utilized in the Internet affect how the Internet functions.	Create a device for sending a single bit of information. Explain or demonstrate how to use their bit sending device to send messages that have more than two states. Develop a protocol for exchanging binary messages in two directions. Explain how synchronization and coordination enable the transmission of binary messages. Define a bit and relate it to the binary messages. Calculate the bit rate for a binary message exchange. Explain why messages need to contain addressing information (sender/recipient identification). Invent an informal addressing protocol for use in the Battleship game. Recall that browsing the Internet entails computers sending each other requests and sending back data to satisfy those requests. Evaluate the benefits and security concerns associated with the use of a routed system of sending packets. Describe the redundancy of routing between two points on the Internet.

	VOCABULARY: ASCII , bit, bit rate, byte, code, IETF, internet, hexadecimal, IP Address, net neutrality, packets, TCP **KEY TERMS:** innovation, protocol, internet	Send messages using a numeric addressing protocol with the Internet Simulator.

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task: (Explore) Prepare a flash talk on a major issue concerning the Internet and its effect on society.
- Fixed Response Assessment modeling AP Style questioning on these topics.

KEY LEARNING EVENTS AND INSTRUCTION:

- Group project: Invent a binary call response protocol, test for speed and accuracy
- Simulate the process of a router joining a network and generating a router table that would allow them to send packets to anyone else in their network as efficiently as possible. Reflect on the process by comparing the similarities between the SSSP problem and the process used and how it facilitates the structure of the Internet.

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT I: The Internet

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
5 Weeks	**The Internet** • **Representing and transferring information** • **Inventing the Internet** • **Performance Task: The Internet and Society**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 CS Principles: Internet Simulator - Part 1 https://youtu.be/Kn6Fd5uwZno The Internet: Packets, Routing and Reliability https://youtu.be/AYdF7b3nMto Code Studio https://studio.code.org/users/sign_in

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT II: Digital Information

STANDARDS / GOALS:	**ENDURING UNDERSTANDINGS**	**ESSENTIAL QUESTIONS**
CSTA K-12 Computer Science Standards **CT - Computational Thinking** CT.L2:14 - Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions. CT.L3A:6 - Analyze the representation and trade-offs among various forms of digital information. CT.L3A:7 - Describe how various types of data are stored in a computer system. CT.L2:7 - Represent data in a variety of ways including text, sounds, pictures and numbers. CT.L2:8 - Use visual representations of problem states, structures and data (e.g., graphs, charts, network diagrams, flowcharts).	A variety of abstractions built upon binary sequences can be used to represent all digital data but there are limitations.	• What kinds of limitations does the binary encoding of information impose on what can be represented inside today's computers? • How accurately can human experience and perception be captured or reflected in digital information?
	People use computer programs to process information to gain insight and knowledge.	• What is the relationship between data, information and knowledge?
	Computing enhances communication, interaction, and cognition by analyzing data and making predictions.	• What are the best ways to find, see, and extract meaningful trends and patterns from raw data?
	The personal aspect of computing has a global affect -- both beneficial and harmful -- on people and society.	• Where and how does human bias affect the collection, processing and interpretation of data?
	KNOWLEDGE	**SKILLS**
	Students will know: At the lowest level, all digital data are represented by bits and at a higher level, bits are grouped to represent abstractions, including but not limited to numbers, characters, and color.	**Students will be able to:** Identify and compare the size of familiar digital media.

CT.L2:9 - Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. CT.L3A:6 - Analyze the representation and trade-offs among various forms of digital information. **Computing Practice and Programming** CPP.L2:4 - Demonstrate an understanding of algorithms and their practical application. CD **- Computers & Communication Devices** CD.L2:4 - Use developmentally appropriate, accurate terminology when communicating about technology. **Computer Science Principles** 2.1 - A variety of abstractions built upon binary sequences can be used to represent all digital data. 3.3 - There are trade-offs when representing information as digital data.	Data representation, storage, security, and transmission involve computational manipulation of information. Digital images can be created by generating pixel patterns, manipulating existing digital images, or combining images. We can use computers to process information, find patterns, and test hypotheses about digitally processed information to gain insight and knowledge .	Use appropriate terminology when describing the size of digital files. Solve small word problems that require reasoning about file sizes. Explain why the optimal amount of compression is impossible or “hard” to identify. Collaborate with a peer to find a solution to a text compression problem using the Text Compression Widget (lossless compression scheme). Develop a strategy (heuristic algorithm) for compressing text. Explain how images are encoded with pixel data. Encode a B&W image in binary representing both the pixel data (intensity) and metadata (width, height). Create the necessary metadata to represent the width and height of a digital image, using a computational tool. Explain why image width and height are metadata for a digital image. Use the Pixelation Tool to encode small color images with varying bits-per-pixel settings. Explain the color encoding scheme for digital images.

1.1 Processes to create computational artifacts for creative expression or to solve a problem. 1.3 - Computing can extend traditional forms of human expression and experience.	Technology enables the collection, use, and exploitation of information about, by, and for individuals, groups, and institutions. Human capabilities are enhanced by digitally enabled collaboration. VOCABULARY: image, metadata, pixel, RGB, aggregation, summary table, KEY TERMS: heuristic, lossless compression, lossy compression, pivot table	Use the Pixelation Tool to encode an image of the student's design. Explain the benefits of using hexadecimal numbers for representing long streams of bits. Describe sources of data appropriate for performing computations. Develop a hypothesis about student behavior over time, based on a small sample of data. Select the appropriate type of data visualization to discover trends and patterns within a dataset. Create a bar, line, and scatter chart from a dataset using a computational tool. Use the settings of a data visualization tool to manipulate and refine the features of a data visualization.

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task: (Create) Analyze the data that they have been collecting as a class in order to demonstrate their ability to discover, visualize, and present a trend or pattern they find in the data

KEY LEARNING EVENTS AND INSTRUCTION:

- Group project: Use the Google Trends tool in order to visualize historical search data. Identify interesting trends or patterns in their findings and will attempt to explain those trends, based on their own experience or through further research online
- Learn about pixels, raster images, and what an image file format is. Encode binary image data using a widget in Code Studio

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT II: Digital Information

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
6 Weeks	**Digital Information** • **Representing and transferring information** • **Inventing the Internet** • **Performance Task: The Internet and Society**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 CS Principles: Internet Simulator - Part 1 https://youtu.be/Kn6Fd5uwZno The Internet: Packets, Routing and Reliability https://youtu.be/AYdF7b3nMto Code Studio https://studio.code.org/users/sign_in

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT III: Algorithms and Programming

STANDARDS / GOALS:	ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
CSTA K-12 Computer Science Standards **CL – Collaboration** CL.L2:2 - Collaboratively design, develop, publish and present products (e.g., videos, podcasts, websites) using technology resources that demonstrate and communicate curriculum. concepts CL.L2:3 - Collaborate with peers, experts and others using collaborative practices such as pair programming, working in project teams and participating in-group active learning activities. **CPP - Computing Practice & Programming** CPP.L2:4 - Demonstrate an understanding of algorithms and their practical application. CPP.L2:8 - Demonstrate dispositions amenable to open-ended problem solving and programming (e.g., comfort with complexity, persistence,	Creative development between many programmers can be an essential process for creating computational artifacts.	• How do programmers collaborate?
	Computing enables people to use creative development processes to create computational artifacts for creative expression or to solve a problem.	• What does it mean to be a “creative” programmer?
	Multiple levels of abstraction are used to write programs or create other computational artifacts	• How can an abstraction help you design a solution for a problem so that it is programmable?
	Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages.	• Why do we need algorithms? • How is designing an algorithm to solve a problem different from other kinds of problem solving?
	KNOWLEDGE	**SKILLS**
	Students will know: Applying a creative development process helps when creating computational artifacts.	**Students will be able to:** Create a set of instructions in human language for building a simple LEGO block arrangement. Assess the clarity of a set of instructions expressed in human language.

brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge). CPP.L3A:3 - Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing) CPP.L3A:4 - Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models). **CT - Computational Thinking** CT.L1:6-01 - Understand and use the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of sample instances, design, implementation and testing). CT.L2:3 - Define an algorithm as a sequence of instructions that can be processed by a computer. CT.L2:1 - Use the basic steps in algorithmic problem-solving to design solutions (e.g., problem	Algorithms are essential in the utility of all programs. Programs implement algorithms to perform a variety of functions.	Describe the ambiguities inherent in human language and the ways programming languages seek to remove those ambiguities. Identify connections between the ability to program and the ability to solve problems. Develop an algorithm to solve a new problem with playing cards. Express an algorithm in the Human Machine Language. Identify Sequencing, Selection and Iteration in a program written the Human Machine Language. Describe the properties of the Human Machine Language that make it a "low level" language. Solve simple programming challenges when the set of allowed commands is constrained. Explain considerations that go into “efficiency” of a program. Use App Lab to write programs that create simple drawings with “turtle graphics.” Work with a partner to program a turtle task that requires about 50 lines of code. Justify or explain choices made when programming a solution to a turtle task. Recognize functions in programs as a form of abstraction.

statement and exploration, examination of sample instances, design, implementing a solution, testing and evaluation). CT.L2:12 - Use abstraction to decompose a problem into sub problems. CT.L3A:1 - Use predefined functions and parameters, classes and methods to divide a complex problem into simpler parts. CT.L3A:3 - Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. CT.L3B:4 - Evaluate algorithms by their efficiency, correctness, and clarity. ***Computer Science Principles*** 4.1 - Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages. 5.2 - People write programs to execute algorithm	There are many levels of abstraction that are used when writing programs. When developing a program, it is sometimes necessary to use algorithms and loops to correctly solve a problem. Writing programs uses functions and parameters.	Write a program that solves a turtle drawing problem using multiple levels of abstraction (i.e. functions that call other functions within your code). Explain why and how functions can make code easier to read and maintain. Define and call simple functions that solve turtle drawing tasks. Write a complete program with functions that solve sub-tasks of a larger programming task. Explain how functions are an example of abstraction. Use a “top-down” problem-solving approach to identify sub-tasks of a larger programming task. Identify appropriate situations in a program for using a loop. Use a loop in a program to simplify the expression of repeated tasks. Use random values within a loop to repeat code that behaves differently each time it is executed. Write functions with parameters to generalize a solution instead of duplicating code. Use random numbers as inputs to function calls for the purpose of testing.

	VOCABULARY: command, function, loop, parameter, program **KEY TERMS:** algorithm, looping	Identify appropriate situations for creating a function with parameters. Add parameters to a function in an existing piece of code to generalize its behavior. Use a loop in a program to simplify the expression of repeated tasks. Use random values within a loop to repeat code that behaves differently each time it is executed. Identify appropriate situations in a program for using a loop.

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task: Design a Digital Scene – design a program that draws a digital scene of their choosing. Students will be working in groups of 3 or 4 and will begin by identifying a scene they wish to create. They will then use Top-Down Design to identify the high-level functions necessary to create that image. The group will then assign these components to individual members of the group to program. After programming their individual portion, students will combine all of their code to compose the whole scene
- Fixed Response Assessment modeling AP Style questioning on these topics

KEY LEARNING EVENTS AND INSTRUCTION:

- Explore the use of functions through a sequence of activities in App Lab.
- Think about the need for functions and how these functions can make their code clearer or more concise
- Write several programs adding new features and increasing in complexity

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT III: Algorithms and Programming

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
6 Weeks	**Algorithms and Programming** • **The need for programming languages and algorithms** • **Creativity in algorithms** • **Using simple commands** • **Creating functions and top down design** • **Functions with parameters** • **Looping**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 Video: You Should Learn to Program: Christian Genco at TEDxSMU https://youtu.be/xfBWk4nw440 Introduction to Functions Tutorial https://youtu.be/yPWQfa4CHbw Code Studio https://studio.code.org/users/sign_in Tutorial: Using Loops https://youtu.be/G8hfAk4PfOM

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT IV: Big Data and Privacy

STANDARDS / GOALS:	**ENDURING UNDERSTANDINGS**	**ESSENTIAL QUESTIONS**
CSTA K-12 Computer Science Standards **CL – Collaboration** CL.L2:2 - Collaboratively design, develop, publish and present products (e.g., videos, podcasts, websites) using technology resources that demonstrate and communicate curriculum. **Computing Practice and Programming (CPP)** 3A-9. Explain the principles of security by examining encryption, cryptography, and authentication techniques. 3B-5. Deploy principles of security by implementing encryption and authentication strategies. 3A-9. Explain the principles of security by examining encryption, cryptography, and authentication techniques. 3B-5. Deploy principles of security	Computing facilitates exploration and the discovery of connections in large amounts of information and allows for a new type of data called "big data".	• How is "big data" different than other types of data we have seen up to this point.?
	An investigative process is aided by effective organization and selection of resources. Appropriate technologies and tools facilitate the accessing of information and enable the ability to evaluate the credibility of sources.	• How can we identify reliable and authoritative sources for our information?
	Cybersecurity is an important concern for the Internet and the systems built on it.	• How is cybersecurity impacting the ever increasing number of internet users?
	Algorithms can solve many but not all computational problems.	• Are the concepts of easy, difficult and impossible problems, when solving algorithmically, subjective?
	KNOWLEDGE	**SKILLS**
	Students will know: Extracting information from data helps discover and explain connections or trends. Large datasets impact the use of computational processes to discover information and knowledge.	**Students will be able to:** Identify sources of data produced, used, and consumed by a web application. Use a tool that provides access to "big data" and investigate its sources. Identify a suitable computing innovation for a research project.

by implementing encryption and authentication strategies		Synthesize information taken from multiple online sources to create a cohesive description of a computing innovation.
Computational Thinking (CT) 3B-2. Explain the value of heuristic algorithms to approximate solutions for intractable problems.		Synthesize information taken from multiple online sources to create a cohesive description of a computing innovation.
3B-3. Critically examine classical algorithms and implement an original algorithm.	Computing has impacted innovations in other fields.	Synthesize information taken from multiple online sources to create a cohesive description of a computing innovation.
3B-4. Evaluate algorithms by their efficiency, correctness, and clarity.	There are many ways in which to evaluate online and print sources for appropriateness and credibility	Evaluate online and print resources for appropriateness and credibility.
Community, Global, and Ethical Impacts (CI) 3A-10. Describe security and privacy issues that relate to computer networks.	Cybersecurity concerns exist but there are potential options to address these issues with the Internet and the systems built on it.	Identify existing cybersecurity concerns and potential options to address these issues with the Internet and the systems built on it
Computer Science Principles 3.3.1 Analyze how data representation, storage, security, and transmission of data involve computational manipulation of information.		Identify existing cybersecurity concerns and potential options to address these issues with the Internet and the systems built on it. Crack or break a message encrypted with a Caesar cipher using the frequency analysis widget.
6.3.1 Identify existing cybersecurity concerns and potential options to address these issues with the Internet and the systems built on it.	There is a difference between solvable and unsolvable problems in computer science when handling the idea of making subjective decisions.	Explain in broad terms what makes a key difficult to “crack.”
7.3.1 Analyze the beneficial and harmful effects of computing.		

	VOCABULARY: big data, encrypt, cipher, frequency analysis, cryptograph **KEY TERMS:** algorithm, encryption `	

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task (Explore): - To conclude their study of cryptography, hard problems, and cybersecurity, students will complete a research project on a cybersecurity innovation of their choosing
- Students will need to identify appropriate online resources to learn about the functionality, context, and impact of their cybersecurity innovation
- After completing their research, students will present their findings both in a written summary and with an audio / visual artifact they found online
- The written components and audio / visual artifact students will identify are similar to those students will see in the AP Performance Tasks.
- Fixed Response Assessment modeling AP Style questioning on these topics

KEY LEARNING EVENTS AND INSTRUCTION:

- Explore how people use “big data” through independent projects to solve problems in many facets of life such as how it aids police officers.
- Investigate a “big data” tool in pairs to evaluate the tool for its usefulness and investigate the source of the data used to make the tool
- Look at the relationship between encryption keys and the passwords used in everyday life.
- Experiment with traits that make a strong password and answer questions about the “human components” of cybersecurity

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
4 Weeks	**Big Data and Privacy** • **What is Big Data** • **Rapid Research: Data Innovations** • **Identifying People with Data** • **The Cost of "Free"** • **The Need for Encrytpion** • **Keys and Passwords**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 Video: Big Data is Better https://www.ted.com/talks/kenneth_cukier_big_data_is_better_data?language=en Impact of Innovations https://www.youtube.com/watch?v=Hg4U8z2OvT8&feature=youtu.be Code Studio https://studio.code.org/s/cspunit4

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT V: Building Apps

STANDARDS / GOALS:	ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
CSTA K-12 Computer Science Standards **Collaboration (CL)** 2-3. Collaborate with peers, experts, and others using collaborative practices such as pair programming, working in project teams, and participating in group active learning activities. 2-4. Exhibit dispositions necessary for collaboration; providing useful feedback, integrating feedback, understanding and accepting multiple perspectives, socialization **Computational Thinking (CT)** 2-6. Describe and analyze a sequence of instructions being followed. 2-12. Use abstraction to decompose a problem into subproblems. 3A-1. Use predefined functions and parameters, classes, and	People write different types of programs depending on their desired event (outcome).	• What makes event-driven programming different from the programs we have written so far?
	Multiple levels of abstraction are used to write programs or create other computational artifacts.	• How does abstraction help us in writing programs, creating computational artifacts, and solving problems?
	Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages.	• Why are some languages better than others when used to implement algorithms?
	KNOWLEDGE	**SKILLS**
	Students will know: A programmer designs, implements, tests, debugs, and maintains programs when solving problems. Code in a programming language is often translated into code in another (lower level) language to be executed on a computer.	**Students will be able to:** Describe an event-driven programming. Provide a general description of asynchronous behavior in event-driven programs. Use a variety of events and screen elements when writing event-driven programs.

methods to divide a complex problem into simpler parts. 3A-3. Explain how sequence, selection, iteration, and recursion are building blocks of algorithms **Computing Practice and Programming (CPP)** 2-5. Implement problem solutions using a programming language, including: looping behavior, conditional statements, logic,expressions, variables, and functions. 3A-3. Use various debugging and testing methods to ensure program correctness. 3A-4. Apply analysis, design, and implementation techniques to solve problems 3A-5. Use APIs and libraries to facilitate programming solutions. **CS Principles Learning Objectives** 2.2.3 Identify multiple levels of abstractions that are used when writing programs. 4.1.1 Develop an algorithm for implementation in a program	Program instructions may involve variables that are initialized and updated, read and written. Programming uses abstraction (including String Data) to manage complexity in programs. (5.3) Selection uses a Boolean condition to determine which of two parts of an algorithm is used.	Explain how the underlying model of event-driven programming remains the same, even when using different page elements and events. Use variables in a program to store numeric values. Store the value returned by a function (randomNumber, promptNum) in a variable for use in a program. Reason about multi-line segments of code in which variables are re-assigned multiple times. Identify strings as a unique data type which contains a sequence of ASCII characters. Manipulate user-generated string input to generate dynamic output. Identify programming problems which require the use of nested or chained conditionals. Write programs that use chained and nested conditional statements. Use functions which return a boolean in the place of a boolean expression.

5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge 5.2.1 Explain how programs implement algorithms.	**VOCABULARY:** event driven program, element, event handler, debug, ASCII, boolean, **KEY TERMS:** event, variable, strings, nested algorithms, arrays	

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task (Create): - To conclude their introduction to programming, students will design an app based on one they have previously worked on in the programming unit. Students will choose the kinds of improvements they wish to make to a past project in order to show their ability to make abstractions and implement complex algorithms. Finally, the project concludes with reflection questions similar to those students will see on the AP Performance Tasks. Students can either complete the project individually or with a partner and every student will need a collaborative partner to give and receive feedback from.

KEY LEARNING EVENTS AND INSTRUCTION:

- Develop a simple user interface and defining its behavior through the use of event-driven programming
- Place buttons on the screen and define their behavior through the use of event handlers
- Become familiar with the logic operators NOT, AND, and OR as tools for simplifying the expression of complex boolean conditions
- Learn how to improve nested or chained conditionals through the use of these logical operations and then use them to make additions to the functionality of the Movie Bot

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT V: Building Apps

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
7 weeks	**Building Apps** • **Event Driven Programming and Debugging** • **Introducing Design Mode** • **Multi-Screen Apps** • **Using Variables in Apps** • **User Inputs and Strings** • **Introduction to Conditional Logic** • **Compound Conditional Logic** • **Looping and Simulation** • **Arrays**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 Video: Introduction to Variables https://www.youtube.com/watch?v=G41G_PEWFjE&feature=youtu.be Events Unplugged https://www.youtube.com/watch?v=CP3ysmT2_PU&feature=youtu.be Code Studio https://studio.code.org/s/cspunit3

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT VI: Performance Tasks

STANDARDS / GOALS:	ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
CSTA K-12 Computer Science Standards CT **- Computational Thinking** CT.L2:14 - Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions. CT.L3A:6 - Analyze the representation and trade-offs among various forms of digital information. CT.L3A:7 - Describe how various types of data are stored in a computer system. CT.L2:7 - Represent data in a variety of ways including text, sounds, pictures and numbers. CT.L2:8 - Use visual representations of problem states, structures and data (e.g., graphs, charts, network diagrams, flowcharts).	An investigative process is aided by effective organization and selection of resources.	• What characteristics of your resources do you think qualify them as "high quality"?
	The AP test results provide an opportunity for colleges to assess the level of college work a student has mastered in high school.	• What learned methods can you apply to show proficiency in the explore and Create Performance Tasks (PTs)?
	KNOWLEDGE	**SKILLS**
	Students will know: The application of the concepts found in computer science in the world today is only limited to the creativity of the programmer. Appropriate technologies and tools facilitate the accessing of information and enable the ability to evaluate the credibility of sources	**Students will be able to:** Prepare for the Create Performance Task by seeking answers to unknown questions. Create and submit the Create Performance Task. Prepare for the Explore Performance Task by seeking answers to unknown questions. Create and Submit the Explore Performance Task.

CT.L2:9 - Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. CT.L3A:6 - Analyze the representation and trade-offs among various forms of digital information. **Computing Practice and Programming** CPP.L2:4 - Demonstrate an understanding of algorithms and their practical application. CD **- Computers & Communication Devices** CD.L2:4 - Use developmentally appropriate, accurate terminology when communicating about technology. **Computer Science Principles** 2.1 - A variety of abstractions built upon binary sequences can be used to represent all digital data. 3.3 - There are trade-offs when representing information as digital data.		

1.1 Processes to create computational artifacts for creative expression or to solve a problem. 1.3 - Computing can extend traditional forms of human expression and experience.		

ASSESSMENT EVIDENCE: Students will show their learning by:

- Performance Task: (Create and Explore) Students will use their knowledge and skills they have developed throughout the year and compete both the explore and create AP tasks and submit them online via the AP Computer Science Principles website
- Fixed Response Assessment modeling AP Style questioning on all learned topics will be used to better prepare the students for the AP exam

KEY LEARNING EVENTS AND INSTRUCTION:

- Use the prior knowledge gained through practice PT’s throughout the year to complete both required PT’s by the AP College Board

AP Computer Science Principles
UNIT VI: Performance Task

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
5 Weeks	**Performance Task and AP Test Review** • **Preparing for Create PT** • **Create PT** • **Preparing for Explore PT** • **Explore PT**	Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008 CS Principles: Internet Simulator - Part 1 https://youtu.be/Kn6Fd5uwZno The Internet: Packets, Routing and Reliability https://youtu.be/AYdF7b3nMto Code Studio https://studio.code.org/

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT VII: Post AP exam Projects and Programming

STANDARDS / GOALS:	**ENDURING UNDERSTANDINGS**	**ESSENTIAL QUESTIONS**
CSTA K-12 Computer Science Standards **CL – Collaboration** CL.L2:2 - Collaboratively design, develop, publish and present products (e.g., videos, podcasts, websites) using technology resources that demonstrate and communicate curriculum. concepts CL.L2:3 - Collaborate with peers, experts and others using collaborative practices such as pair programming, working in project teams and participating in-group active learning activities. **CPP - Computing Practice & Programming** CPP.L2:4 - Demonstrate an understanding of algorithms and their practical application. CPP.L2:8 - Demonstrate dispositions amenable to open-ended problem solving and	Computers are problem solving machines that are only limited by their programmers.	• How does a programmer use a computer program to get a computer to perform a task?
	Object-oriented programming helps programmers write software solutions in Java.	• How can a real-world object be represented as a Java class?
	KNOWLEDGE	**SKILLS**
	Students will know: A computer is comprised of hardware and software components. The goal of software design is to correctly solve a problem using a program that can be adapted to changing circumstances.	**Students will be able to:** Identify the hardware components of a computer including CPU, RAM, ROM, hard drive, power supply, motherboard, peripherals, ports, network card. Solve logic problems using the basic types of logic gates including AND, OR, and XOR. Explain the differences between machine language, assembly language, and middle and higher level programming level languages. Translate back and forth between decimal, hexadecimal, and binary number systems. Download and learn to use either the BlueJ or Eclipse IDE.

programming (e.g., comfort with complexity, persistence, brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge). CPP.L3A:3 - Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing) CPP.L3A:4 - Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models). **CT - Computational Thinking** CT.L1:6-01 - Understand and use the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of sample instances, design, implementation and testing).	An integrated development environment (IDE) is a program that allows programmers to develop and debug programs and applications. A class consists of instance variables and methods. The use of inheritance to create an "is-a" relationship and a "has-a" relationship. Vocabulary: CPU (Central Processing Unit), logic gates, memory, peripherals, software, internet, compiler	Create, debug and compile a simple java project in the IDE. Explain what compilers and interpreters do and discuss how bytecode and the Java Virtual Machine (JVM) are related. Design a simple class in the IDE including: • public and private instance variable • public and private methods Explain why well-designed classes consider the principle of information-hiding. Break a problem down into smaller, more manageable parts. Determine an objects state and behavior. Design simple classes which inherit the properties of other classes.

CT.L2:3 - Define an algorithm as a sequence of instructions that can be processed by a computer. CT.L2:1 - Use the basic steps in algorithmic problem-solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing and evaluation). CT.L2:12 - Use abstraction to decompose a problem into sub problems. CT.L3A:1 - Use predefined functions and parameters, classes and methods to divide a complex problem into simpler parts. CT.L3A:3 - Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. CT.L3B:4 - Evaluate algorithms by their efficiency, correctness, and clarity.		

Computer Science Principles 4.1 - Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages. 5.2 - People write programs to execute algorithm		

ASSESSMENT EVIDENCE: Students will show their learning by:

- Project: Students will transition into basic Programming outside of the block style programming used within the AP Computer Science Principles course
- Using Java Students will create simple playable games and present them to the class for testing and critique

KEY LEARNING EVENTS AND INSTRUCTION:

- Explore 2 main ideas
    1) Dissect the hardware of old computers in order to better understand how each component functions in conjunction with the others
    2) Transition from block/app programming into Java programming as they work with Computer Science A students on the basics of the Java/C++/python languages

RANDOLPH TOWNSHIP SCHOOL DISTRICT
AP Computer Science Principles
UNIT VII: Post AP Exam Projects and Programming

SUGGESTED TIME ALLOTMENT	CONTENT-UNIT OF STUDY	SUPPLEMENTAL UNIT RESOURCES
5 Weeks	**Post AP Exam Projects and Programming** • **Introduction to Software, Hardware, Internet** • **Software Development** • **Compilers and Interpreters** • **Software Packages** • **Object-oriented programming** • **Objects and Classes** • **Objects and Classes** • **Encapsulation**	Integrated Development Environment: http://www.eclipse.org Integrated Development Environment: http://www.BlueJ.org Java Platform, Standard Edition 7, API Specification: docs.oracle.com/javase/7/docs/api/ Online Java Coding Practice: codingbat.com Software to learn Java programming creating simple games.: www.greenfoot.com

APPENDIX A

Sample Unit Plan Outline
Randolph High School
AP CSP
“The Internet”

Subject: AP Computer Science Principles

Unit Topic and Length:

- **The Internet:** This unit begins exploring the technical challenges and questions that arise from the need to represent digital information in computers and transfer it between people and computational devices. In the second half of the unit, students solve problems similar ones that had to be solved to build the real Internet. Students design their own versions of protocols, each one layered on the previous one, in a process that mimics the layered sets of protocols on the real Internet. Topics include: the digital representation of numbers and text, Internet Protocol, DNS, and TCP/IP.
- Unit will take approx.. 5 weeks

Common Core Content Standards:

CSTA K-12 Computer Science Standards

CI.L3B:2,4

CL.L2:3,4

CT.L2:7,8,9,14

Computer Science Principles
2.1, 2.3 ,3.1

Enduring Understandings:

- There are trade-offs when representing information as digital data.
- Characteristics of the Internet influence the systems built on it.
- A variety of abstractions built upon binary sequences can be used to

Essential Questions:

- Is binary the best way to represent digital information?
- Are the ways data is represented and transmitted with computers laws of nature or a laws of man?
- Who and what is “in charge” of the Internet and how it functions?

represent all digital data on the internet.	➢ In what ways are you or your family directly affected by computers and/or computer science ?
CONTENT: ➢ Computing innovations have impacted innovations in other fields and help to analyze the beneficial and harmful effects of computing. ➢ There are a variety of abstractions used to represent data. ➢ Data representation, storage, security, and transmission of data involve computational manipulation of information. ➢ The internet has a set of specific characteristics which are essential to the systems built on it.	**SKILLS:** ➢ Describe positive and negative effects of computing innovations. ➢ Communicate with classmates about computing innovations in their lives. ➢ Analyze the possibilities and limitations that arise when sending binary messages ➢ Create a device for sending a single bit of information. ➢ Explain or demonstrate how to use their bit sending device to send messages that have more than two states. ➢ Develop a protocol for exchanging binary messages in two directions. ➢ Explain how synchronization and coordination enable the transmission of binary messages. ➢ Define a bit and relate it to the binary messages. ➢ Calculate the bit rate for a binary message exchange. ➢ Explain why messages need to contain addressing information (sender/recipient identification). ➢ Invent an informal addressing protocol for use in the Battleship game.
VOCABULARY/KEY TERMS: ➢ ASCII , bit, bit rate, byte, code, IETF, Internet, hexadecimal, IP Address, Net Neutrality, Packets, TCP, Innovation, protocol, Internet	

ASSESSMENT EVIDENCE AND ACTIVITIES:

- Performance Task: (Explore) Prepare a flash talk on a major issue concerning the Internet and its effect on society.
- Fixed Response Assessment modeling AP Style questioning on these topics.

RESOURCES: RESOURCES:

Abelson, Al, Ledeen, Ken, and Lewis, Harry, *Blown to Bits: Your Life, Liberty, and Happiness After the Digital Explosion*, Boston, Mass - Addison Wesley, 2008

CS Principles: Internet Simulator - Part 1
https://youtu.be/Kn6Fd5uwZno

The Internet: Packets, Routing and Reliability
https://youtu.be/AYdF7b3nMto

Code Studio
https://studio.code.org/users/sign_in

AP Computer Science Principles Performance Task

Create — Applications from Ideas

Overview

Programming is a collaborative and creative process that brings ideas to life through the development of software. Programs can help solve problems, enable innovations, or express personal interests. In this Performance Task, you will be developing a program of your choice. Your development process should include iteratively designing, implementing, and testing your program. You are strongly encouraged to work with another student in your class.

Please note that once this performance task has been assigned as an assessment (rather than as practice), you are expected to complete the task with minimal assistance from anyone other than your collaborative partner. For more clarification see the AP Computer Science Principles Assessment Overview for Students found in the Course and Exam Description.

You will be provided with 12 hours of class time to complete and submit the following:

- **A video of your program running**
- **Written responses about your program and development process**
- **Program Code**

Your teacher will share submission guidelines that include suggestions for creating video and PDF files.

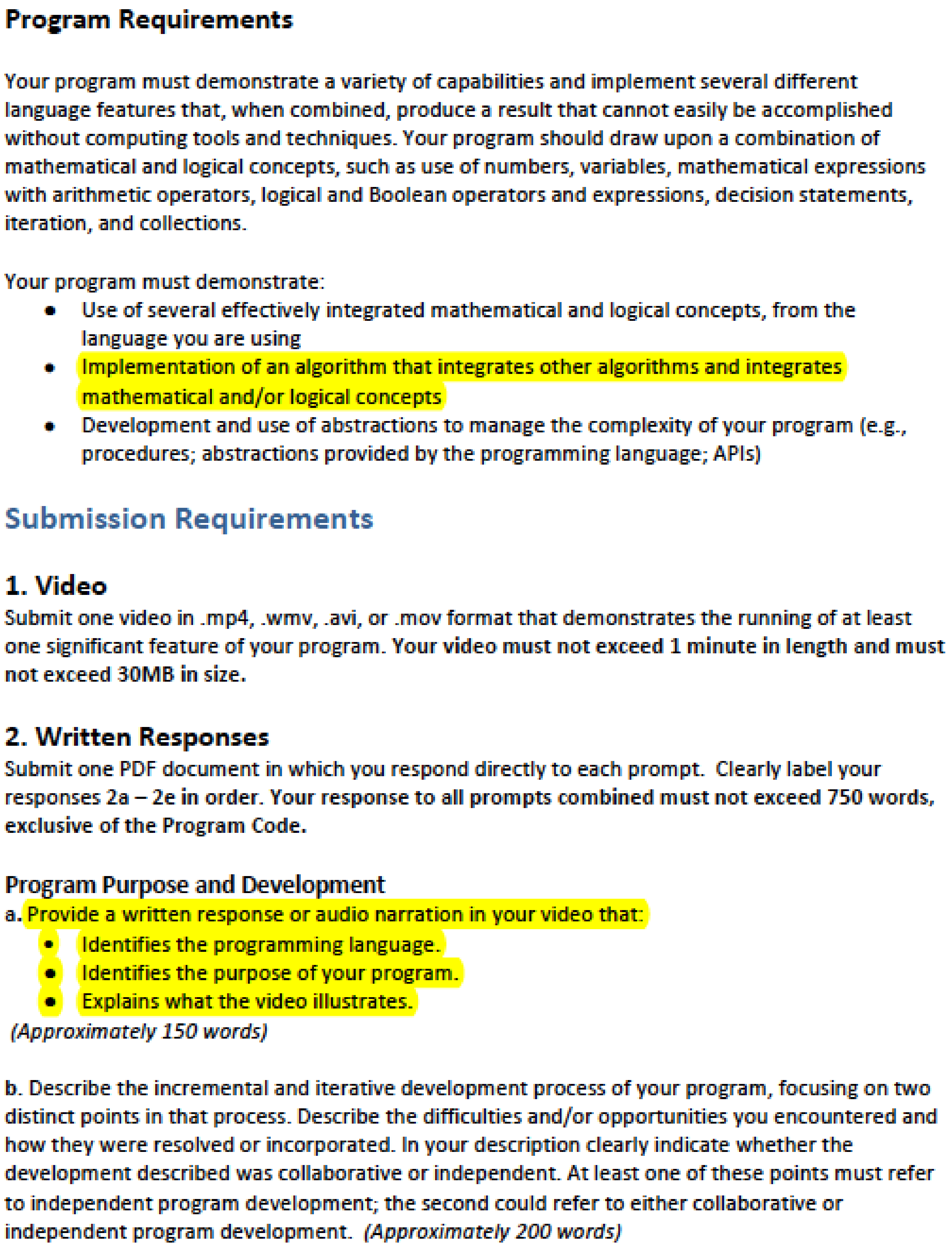
Note: Students in nontraditional classroom environments should consult a school-based AP Coordinator for instructions.

General Requirements

This performance task requires you to develop a program on a topic that interests you or one that solves a problem. It is strongly recommended that a portion of the program involve some form of collaboration with another student in your class. Your program development process must involve a significant portion of work completed independently that requires a significant level of planning, designing, and program development.

You are required to:

- Iteratively design, implement, and test your program.
- Independently create at least one significant part of your program.
- Create a video that displays the running of your program and demonstrates its functionality.
- Write responses to questions about your program.
- Include your entire program code.



Program Requirements

Your program must demonstrate a variety of capabilities and implement several different language features that, when combined, produce a result that cannot easily be accomplished without computing tools and techniques. Your program should draw upon a combination of mathematical and logical concepts, such as use of numbers, variables, mathematical expressions with arithmetic operators, logical and Boolean operators and expressions, decision statements, iteration, and collections.

Your program must demonstrate:

- Use of several effectively integrated mathematical and logical concepts, from the language you are using
- Implementation of an algorithm that integrates other algorithms and integrates mathematical and/or logical concepts
- Development and use of abstractions to manage the complexity of your program (e.g., procedures; abstractions provided by the programming language; APIs)

Submission Requirements

1. Video

Submit one video in .mp4, .wmv, .avi, or .mov format that demonstrates the running of at least one significant feature of your program. **Your video must not exceed 1 minute in length and must not exceed 30MB in size.**

2. Written Responses

Submit one PDF document in which you respond directly to each prompt. Clearly label your responses 2a – 2e **in order. Your response to all prompts combined must not exceed 750 words, exclusive of the Program Code.**

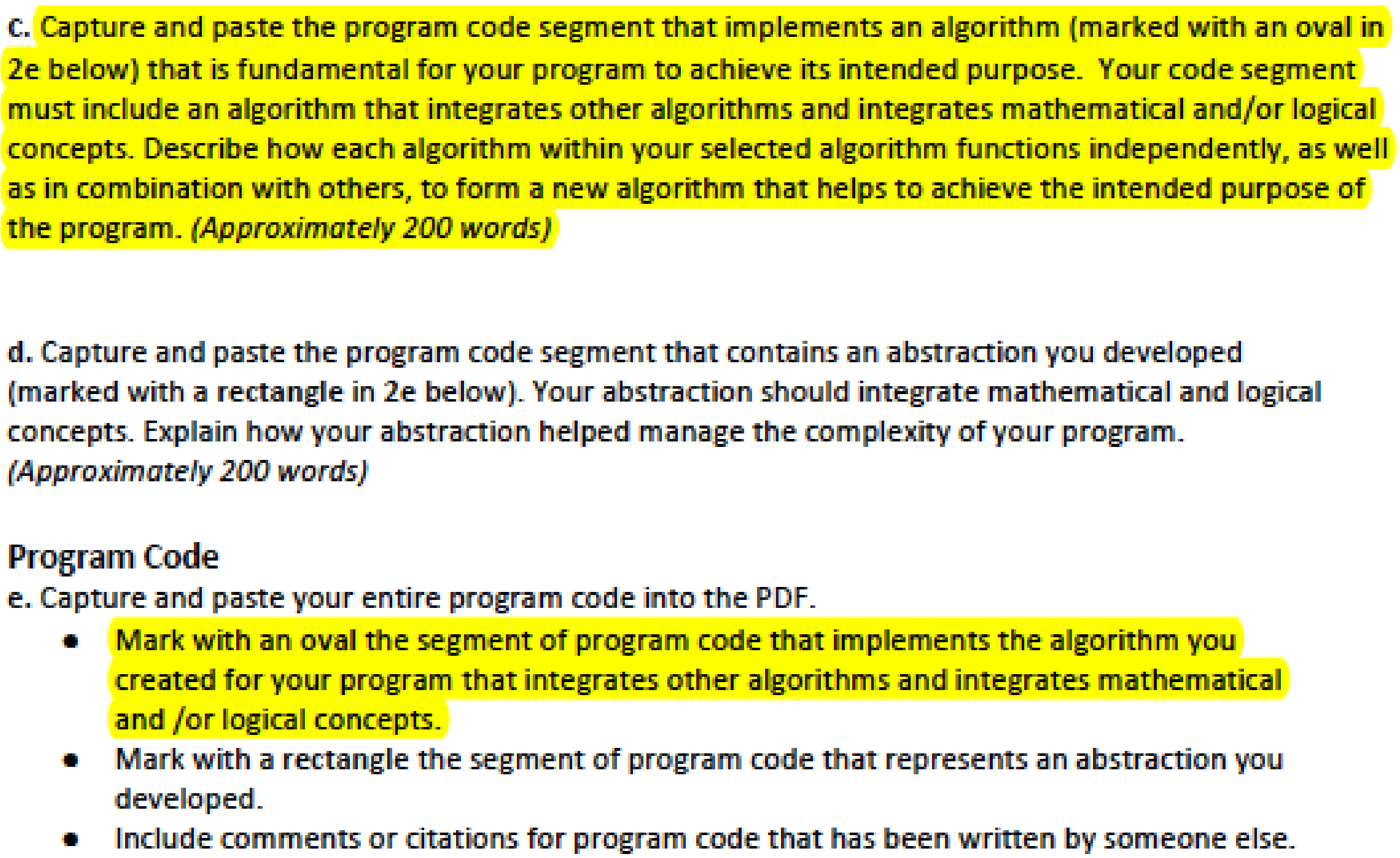
Program Purpose and Development

a. Provide a written response or audio narration in your video that:

- Identifies the programming language.
- Identifies the purpose of your program.
- Explains what the video illustrates.

(Approximately 150 words)

b. Describe the incremental and iterative development process of your program, focusing on two distinct points in that process. Describe the difficulties and/or opportunities you encountered and how they were resolved or incorporated. In your description clearly indicate whether the development described was collaborative or independent. At least one of these points must refer to independent program development; the second could refer to either collaborative or independent program development. *(Approximately 200 words)*



c. Capture and paste the program code segment that implements an algorithm (marked with an oval in 2e below) that is fundamental for your program to achieve its intended purpose. Your code segment must include an algorithm that integrates other algorithms and integrates mathematical and/or logical concepts. Describe how each algorithm within your selected algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program. *(Approximately 200 words)*

d. Capture and paste the program code segment that contains an abstraction you developed (marked with a rectangle in 2e below). Your abstraction should integrate mathematical and logical concepts. Explain how your abstraction helped manage the complexity of your program. *(Approximately 200 words)*

Program Code

e. Capture and paste your entire program code into the PDF.

- Mark with an oval the segment of program code that implements the algorithm you created for your program that integrates other algorithms and integrates mathematical and /or logical concepts.
- Mark with a rectangle the segment of program code that represents an abstraction you developed.
- Include comments or citations for program code that has been written by someone else.

AP Computer Science Principles Performance Task Draft Rubric: *INVESTIGATE*

ASPECT	PERFORMANCE QUALITY			SCORE
Collaborative Report	The dataset represents a rich factual source that supports the discovery of useful, expansive information and knowledge for answering all of the investigative questions. Questions and data focus on significant properties and multiple attributes, leading to meaningful discovery of new information or knowledge. Questions are insightful and lead to multiple connections among data, information, and knowledge. Non-textual representations, if present, created by tool(s) communicate insight and knowledge gained through analysis and enhance the answers to the questions posed. The report provides clear insights into how the questions raised and answers generated contribute knowledge and insight to the identified field.	There is reasonable alignment between the data set and the questions. Combined, the questions and data set enable the discovery of useful information and knowledge that helps answer some portion of the investigative questions. Questions and data focus on a limited range of attributes, limiting the scope of meaningful discovery of new information or knowledge. Questions lead to some useful connections among data, information, and knowledge. Minor discrepancies exist between the answers to the questions posed and any accompanying non-textual representations. The report contributes knowledge to the identified field. While going beyond mere description, the report provides little relevant insight.	The data set is too small to support the generation of information and knowledge. The connection between the data set and the accompanying questions provides a weak foundation for answering the investigative questions. Questions and data focus on the properties of a very limited range of attributes, greatly limiting the scope of meaningful discovery of new information or knowledge. Questions are superficial and can be answered using limited data with few, if any, connections among data, information, and knowledge. Non-textual representations, if present, provide minimal insight and knowledge gained from tool use and are misaligned with the textual answers to the questions posed. The report delivers confusing or misleading information and makes weak connections between knowledge gained during the investigation and the identified field.	
	5-4	**3-2**	**1-0**	
Individual Report: Use of computational tools and techniques	Use of the selected tools and techniques demonstrates strategic and creative processing of information to produce meaningful artifacts. Computational techniques used to manipulate large data sets enable extensive, meaningful exploration and discovery of connections among data, information and knowledge.	Use of the selected tools and techniques demonstrates able processing of information to produce artifacts. Computational techniques used to manipulate large data sets enable some meaningful exploration and discovery of connections among data, information and knowledge.	Use of the selected tools and techniques demonstrates little processing of information to produce artifacts. The limited size of the data set and/or the techniques used to manipulate the data result in little or no exploration and discovery of connections among data, information and knowledge.	
	5-4	**3-2**	**1-0**	

Individual Report: Acquisition of insight and knowledge	The investigation's purposeful use of computing is directly related to the resulting insight and knowledge. The process and tools used promote a depth of understanding. The description is such that it allows for the replication of the investigation and verification of results.	The investigation's use of computers demonstrates the transformation of data into information and knowledge. Additional information is required to fully understand or replicate the investigation, based on the narrow description of the process and tools used.	The investigation's use of computers minimally contributes to the development of insight and knowledge. The analysis of the processes and tools used to manage information lacks focus.	
	5-4	**3-2**	**1-0**	
Individual Reflection	The reflection on collaboration demonstrates a high level of cooperation between partners in coordinating the workload. The partners communicate effectively in sharing what each knows, questioning each other and giving feedback, and reviewing and revising their work.	The reflection on collaboration demonstrates a balance between partners in coordinating the workload. The partners communicate in sharing what they know. However, they skip the important steps of providing each other with feedback and reviewing and revising their work.	The reflection on collaboration indicates that the partner work is primarily independent, each contributing separate portions of an artifact without receiving insights or feedback from each other.	
	5-4	**3-2**	**1-0**	
			Total Score – INVESTIGATE (max = 20)	

AP Computer Science Principles Performance Task Draft Rubric: *EXPLORE*

ASPECT	PERFORMANCE QUALITY			SCORE
Digital Artifact	The artifact provides significant insight into the chosen area of interest and demonstrates a high level of creativity. It was generated using an effective combination of the available features of the chosen computational tool(s). Creation of the artifact is strongly supported by the necessary use of the computational tool(s).	The artifact provides some insight into the chosen areas of interest and demonstrates creativity. It was generated using a combination of the available features of the chosen computational tools. Creation of the artifact is supported by the necessary use of the computational tool(s).	The artifact provides little insight into the chosen area of interest. It was created using limited, mostly unrelated features of the chosen computational tool(s).The artifact can be created manually.	
	5-4	**3-2**	**1-0**	
Written Report: Identification of, and analysis of impact on, a given population	The submission precisely identifies a significant population that has been affected. It presents a rich analysis of the computing innovation and describes in detail how it affects the identified population economically, socially, or culturally. High-quality referenced sources provide context to the examined innovation. The analysis demonstrates a strong understanding of the impact of the innovation within a population. Impact includes how the innovation affects communication, interaction, and cognition within a population.	The submission generally identifies a significant population that has been affected. It presents an analysis of the computing innovation and identifies how it affects the identified population economically, socially, or culturally. The referenced sources provide context to the examined innovation. The analysis demonstrates a clear relationship between the innovation and its effects on communication, interaction, or cognition within a population.	The submission minimally or partially identifies an affected population of a small size. It loosely connects the innovation to the identified economic, social, or cultural impact. The sources lack proper references and align with a context other than the one identified (economic, social, or cultural). The analysis demonstrates a weak relationship between the examined innovation and its effects on communication, interaction, or cognition within a population.	
	5-4	**3-2**	**1-0**	
Written Report: Explanation of algorithms and abstractions	A clear explanation of at least one algorithm and one use of abstraction associated with the innovation are included.	An explanation of at least one algorithm associated with the innovation is included.	No explanation of any algorithm or use of abstraction associated with the innovation is included.	
	5-4	**3-2**	**1-0**	
Written Report: Analysis of beneficial and harmful effects	The analysis offers a clear, well-supported explanation of both beneficial and harmful impacts of the innovation. Security and privacy issues associated with the innovation are effectively analyzed and presented in the report.	The analysis offers a partial explanation of the innovation's beneficial and harmful impacts. Security and privacy issues associated with the innovation have been identified in the report.	The analysis disregards the beneficial and/or harmful impacts of the innovation, including security and privacy issues.	
	5-4	**3-2**	**1-0**	
Total Score – EXPLORE (max = 20)				

Practice PT Overview and Rubric - Design a Digital Scene

Overview

You will submit this project and write responses to the reflection questions in the style of the AP® Create Performance Task. The document below has been constructed to mimic the AP Create Performance Task. Some but not all of the language is pulled directly from the AP document. Some of the prompts have been modified slightly or simply omitted for clarity and to better fit the *Design a Digital Scene* project.

Programming Requirements

Process

The process of creating your program includes individual and collaborative work. Individual work means some portions of the design, development, and implementation of your program must be completed independently. Collaboration can take different forms and can occur at different times in the program development process.

You will be required to respond to prompts about your collaboration, as well as to identify the portions of your program that were created independently. The following are examples of different forms of collaboration:

- **A.** Collaboration can take the form of brainstorming and sharing ideas before the process of writing code begins. Partners can then choose to work together or independently at selected times during the programming process.
- **B.** Collaboration can take the form of working together to develop an idea, beginning the programming process together, and then working independently to add different features to the collaboratively-developed portion of the program.
- **C.** Collaboration can involve Pair Programming, in which one partner “drives” (enters code) while the other “navigates” (recommends and reviews code entered by driver), with the partners changing roles after designated time intervals.
- **D.** Collaboration can involve each partner developing pieces of the program and combining those pieces during the development process.
- **E.** Collaboration can blend any or all of the above techniques and may include an iterative process in which one or more of these techniques, or other collaboration techniques, are employed several times in the program design, development, and testing phases.

Program

Your program must demonstrate a variety of capabilities and implement several different language features that, when combined, produce a result that cannot be easily accomplished without computing tools and techniques. **You will be required to respond to prompts about the program development process and your program code, including questions about the abstractions you used. The program must demonstrate:**

- Use of several effectively integrated programming elements from the programming language you are using

- Use and creation of abstractions to manage the complexity of your program (e.g., functions/procedures; abstractions provided by the programming language; APIs)

Submission Requirements

1. Group Planning Document

As a group, you will submit a **single copy** of your group planning document. Make sure that all group members' names are listed on the document.

2. Program Code

When you have completed your digital scene, submit it to your teacher by clicking the "Submit Project" button on the proper App Lab project in Code Studio (the project associated with this lesson). Submitting indicates to the teacher that your project is ready to be reviewed.

Your final digital scene **code should:**

- contain the functions you wrote and the functions you got from your teammates
- make calls to both the functions you wrote and your teammates' functions in your program.

Note: It is OK if you had to alter your teammates' code slightly to make it work in your program, or to improve its functionality.

3. Individual Written Responses

After completing your project, respond to each of the following reflection questions. **Your response to any one prompt must not exceed 300 words.**

a. Provide an overview of the purpose of your program and how your program code works. Describe the most important program features, rather than providing a line-by-line summary of the program code.

b. Describe the most difficult programming problem you encountered while writing your individual code. What was the difficulty? Explain how you resolved it.

c. Identify an abstraction used in your program and explain how it helped manage the complexity of your program.

d. Explain in detail points in your development process where collaboration was used.
   - Describe the form of collaboration you used. Refer to Process section A-E in your description.
   - Explain how this collaboration affected your program development. Cite specific examples from the collaboration, such as how the group worked together to arrive at solutions, or feedback that you gave and received.

Rubric - Design a Digital Scene

Component	1	2	3	Score
Group Planning Document				
Project Design	The description and/or sketch/digital image of the design are simplistic and lacking in details. Not enough information is given to realistically build a program from.	The description and/or sketch/digital image are limited in details. While it might be possible to program from the design, there are too many details missing for the programming task to be easy.	The description and/or sketch/digital image are rich in details. A programmer would have few questions and find it easy to work from this design.	
Top-Down Design	The image has not been broken into logical components, or most components have not been assigned a high-level function with a descriptive name.	Most aspects of the image have been broken into components; however, the components are not distinct or logical and do not represent top-down design, or the functions are poorly named.	The image has been broken into logical components that represent top-down design; each component has been assigned a high-level function with a descriptive name.	
Task Assignments	Tasks have not been evenly divided among team members and/or tasks have not been prioritized.	Tasks have been divided among members but the assignments and prioritizations do not reflect a realistic estimate of the time constraints or anticipate problems that might arise.	Tasks have been prioritized and evenly divided among members with considerations made for timing or problems that might arise.	
Program Code				
Functions and Abstraction	The program does not make use of the high-level functions agreed upon by the team.	The program makes limited or inconsistent use of levels of abstraction and the functions created by the team members.	Appropriate levels of abstraction are expressed in code using the high-level functions created by the team. (Modifications of functions are allowed.)	
Functions with Parameters	The program does not feature a function with a parameter.	Program contains a function with a parameter that is used in a limited or trivial way.	Program contains at least one function with a parameter to control behavior in a meaningful way.	
Loops	The program does not feature a loop.	A loop is used in a limited or trivial way to repeatedly execute portions of code.	A loop is used in a meaningful way to repeatedly execute portions of code.	

Functionality	There is a weak connection between the output of your function(s) and the function descriptions agreed upon by the group.	There is a moderate connection between the output of your function(s) and the function descriptions agreed upon by the group.	There is a clear and obvious connection between the output of your function(s) and the function descriptions agreed upon by the group.	
Final Scene Incorporates Work from Teammates	Final digital scene does not make use of code written by other teammates.	Final digital scene uses code written by some of the other teammates.	Final digital scene uses code written by each of the teammates. (Note: Students may make alterations to their teammates' code as needed to function correctly in the final scene.)	
Individual Written Responses				
a. Program Overview	The connection between the program and its purpose is unclear. Or it is unclear how the program features connect to the purpose.	There is a logical connection between the program and its purpose. Or the purpose of the program is weakly supported by the features identified.	There is a compelling connection between the program and its stated purpose, supported by details of the important features identified.	
b. Difficulties Encountered	The response generally describes the development of the program without clearly describing any specific problems.	The response describes the developmental steps of the program, but it includes little or no information about how any problems were addressed.	The response fully describes the development details that enable the reader to understand the the difficulties that were encountered and how they were resolved.	
c. Abstraction	The explanation of how the selected code illustrates abstraction is incorrect or incomplete.	The explanation of how the selected code illustrates abstraction is mostly complete but lacks a clear explanation of how it helps manage the complexity of the program.	The explanation of how the selected code illustrates abstraction is well-supported by details and clearly describes how it helps manage the complexity of the program.	
d. Collaboration	The response describes a non-collaborative process, or an ineffective collaborative process. The explanation does not describe how the process affected the program development.	The response cites some effective collaboration, but it is unclear how the collaboration affected the program development or any feedback was provided or incorporated.	The response describes effective collaboration and cites specific examples of how collaboration impacted the program development. Examples of providing and incorporating feedback are included.	