Unit 1: Digital Information (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 1**
Length:	**3 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA #2: DATA - Data are central to computing innovations because they communicate initial conditions to programs and represent new knowledge. Computers consume data, transform data, and produce new data, allowing users to create new information or knowledge to solve problems through the interpretation of those data. Computers store data digitally, which means that the data must be manipulated in order to be presented in a useful way to the user.
- BIG IDEA 5: IMPACT OF COMPUTING (IOC) - Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand the potential impacts of our programs and be responsible for the consequences. As computer users, we need to understand any potential beneficial or harmful effects and how to protect ourselves and our privacy when using a computer.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.DC.1	Explain the beneficial and harmful effects that intellectual property laws can have on the creation and sharing of content (e.g., 6.1.12.CivicsPR.16.a).
TECH.9.4.12.DC.4	Explain the privacy concerns related to the collection of data (e.g., cookies) and generation of data through automated processes that may not be evident to users (e.g., 8.1.12.NI.3).
TECH.9.4.12.DC.5	Debate laws and regulations that impact the development and use of software.
TECH.9.4.12.DC.7	Evaluate the influence of digital communities on the nature, content and responsibilities of careers, and other aspects of society (e.g., 6.1.12.CivicsPD.16.a).
TECH.9.4.12.DC.8	Explain how increased network connectivity and computing capabilities of everyday objects allow for innovative technological approaches to climate protection.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).
TECH.9.4.12.TL.3	Analyze the effectiveness of the process and quality of collaborative environments.
TECH.9.4.12.TL.4	Collaborate in online learning communities or social networks or virtual worlds to analyze and propose a resolution to a real-world problem (e.g., 7.1.AL.IPERS.6).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.CS.1	Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
CS.9-12.8.1.12.CS.2	Model interactions between application software, system software, and hardware.
CS.9-12.8.1.12.DA.2	Describe the trade-offs in how and where data is organized and stored.
CS.9-12.8.1.12.DA.3	Translate between decimal numbers and binary numbers.
CS.9-12.8.1.12.DA.4	Explain the relationship between binary numbers and the storage and use of data in a computing device.
CS.9-12.8.2.12.EC.1	Analyze controversial technological issues and determine the degree to which individuals, businesses, and governments have an ethical role in decisions that are made.
CS.9-12.8.2.12.ITH.3	Analyze the impact that globalization, social media, and access to open source technologies has had on innovation and on a society's economy, politics, and culture.

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Transfer Goals and Career Ready Practices

Transfer Goals

Students will be able to independently use their learning to encode, represent and manipulate data.

Concepts

Essential Questions

- Are innovators responsible for the harmful effects of their computing innovations, even if those effects were unintentional? Why or why not?
- How can we use 1s and 0s to represent something complex like a video of the marching band playing a song?
- What app or computer software do you use most often and would have a hard time going without? How does this software solve a problem for you or benefit you?

Understandings

- The way a computer represents data internally is different from the way the data are interpreted and displayed for the user. Programs are used to translate data into a representation more easily understood by people. (DAT-1)
- While computing innovations are typically designed to achieve a specific purpose, they may have unintended consequences. (IOC-1)

Critical Knowledge and Skills

Knowledge

Students will know:

- DAT-1.A.1 Data values can be stored in variables, lists of items, or standalone constants and can be passed as input to (or output from) procedures.
- DAT-1.A.10 Analog data can be closely approximated digitally using a sampling technique, which means measuring values of the analog signal at regular intervals called samples. The samples are measured to figure out the exact bits required to store each sample
- DAT-1.A.2 Computing devices represent data digitally, meaning that the lowest-level components of any value are bits
- DAT-1.A.3 Bit is shorthand for binary digit and is either 0 or 1.
- DAT-1.A.4 A byte is 8 bits.
- DAT-1.A.5 Abstraction is the process of reducing complexity by focusing on the main idea. By hiding details irrelevant to the question at hand and bringing together related and useful details, abstraction reduces complexity and allows one to focus on the idea.
- DAT-1.A.6 Bits are grouped to represent abstractions. These abstractions include, but are not limited to, numbers, characters, and color.
- DAT-1.A.7 The same sequence of bits may represent different types of data in different contexts.
- DAT-1.A.8 Analog data have values that change smoothly, rather than in discrete intervals, over time. Some examples of analog data include pitch and volume of music, colors of a painting, or position of a sprinter during a race.
- DAT-1.A.9 The use of digital data to approximate real world analog data is an example of abstraction.
- DAT-1.D.1 Data compression can reduce the size (number of bits) of transmitted or stored data.
- DAT-1.D.2 Fewer bits does not necessarily mean less information.
- DAT-1.D.3 The amount of size reduction from compression depends on both the amount of redundancy in the original data representation and the compression algorithm applied.
- DAT-1.D.4 Lossless data compression algorithms can usually reduce the number of bits stored or transmitted while guaranteeing complete reconstruction of the original data.
- DAT-1.D.5 Lossy data compression algorithms can significantly reduce the number of bits stored or transmitted but only allow reconstruction of an approximation of the original data.
- DAT-1.D.6 Lossy data compression algorithms can usually reduce the number of bits stored or transmitted more than lossless compression algorithms.
- DAT-1.D.7 In situations where quality or ability to reconstruct the original is maximally important, lossless compression algorithms are typically chosen.
- DAT-1.D.8 In situations where minimizing data size or transmission time is maximally important, lossy compression algorithms are typically chosen.
- OC-1.F.1 Material created on a computer is the intellectual property of the creator or an organization.
- OC-1.F.10 The digital divide raises ethical concerns around computing.
- OC-1.F.11 Computing innovations can raise legal and ethical concerns.
- OC-1.F.2 Ease of access and distribution of digitized information raises intellectual property concerns regarding ownership, value, and use
- OC-1.F.3 Measures should be taken to safeguard intellectual property
- OC-1.F.4 The use of material created by someone else without permission and presented as one’s own is plagiarism and may have legal consequences.
- OC-1.F.5 Some examples of legal ways to use materials created by someone else include: § Creative Commons—a public copyright license that enables the free distribution of an otherwise copyrighted work. This is used when the content creator wants to give others the right to share, use, and build upon the work they have created. § open source—programs that are made freely available and may be

redistributed and modified § open access—online research output free of any and all restrictions on access and free of many restrictions on use, such as copyright or license restrictions

- OC-1.F.6 The use of material created by someone other than you should always be cited.
- OC-1.F.7 Creative Commons, open source, and open access have enabled broad access to digital information.
- OC-1.F.8 As with any technology or medium, using computing to harm individuals or groups of people raises legal and ethical concerns.
- OC-1.F.9 Computing can play a role in social and political issues, which in turn often raises legal and ethical concerns.

Skills

Students will be able to:

- 1.D Evaluate solution options.
- 2.B Implement and apply an algorithm.
- 3.C Explain how abstraction manages complexity.
- 5.E Evaluate the use of computing based on legal and ethical factors.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- Code Studio Review and Reflection Questions
- Code.org Activity Guides
- Submission of Google Sheets

School Summative Assessment Pan

- Project - Digital Information Dilemmas
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/courses/csp-2022)

Supplementary Resources

- Blown to Bits (http://www.bitsbook.com/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❏ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❏ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❏ All assignments have been created in the student's native language.

❏ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Learning Plan / Pacing Guide

WEEK 1:

- Lesson 1: Welcome to CSP
- Lesson 2: Representing Information
- Lesson 3: Circle Square Patterns
- Lesson 4: Binary Numbers

WEEK 2:

- Lesson 5: Overflow and Rounding
- Lesson 6: Representing Text
- Lesson 7: Black and White Images
- Lesson 8: Color Images
- Lesson 9: Lossless Compression

WEEK 3:

- Lesson 10: Lossy Compression
- Lesson 11: Intellectual Property
- Lesson 12-13: Project - Digital Information Dilemma
- Lesson 14: Unit Assessment

Unit 2: The Internet (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 1**
Length:	**2 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA 4: COMPUTING SYSTEMS AND NETWORKS (CSN) Computer systems and networks are used to transfer data. One of the largest and most commonly used networks is the Internet. Through a series of protocols, the Internet can be used to send and receive information and ideas throughout the world. Transferring and processing information can be slow when done on a single computer, but leveraging multiple computers to do the work at the same time can significantly shorten the time it takes to complete tasks or solve problems.
- BIG IDEA 5: IMPACT OF COMPUTING (IOC) Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand the potential impacts of our programs and be responsible for the consequences. As computer users, we need to understand any potential beneficial or harmful effects and how to protect ourselves and our privacy when using a computer.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.DC.1	Explain the beneficial and harmful effects that intellectual property laws can have on the creation and sharing of content (e.g., 6.1.12.CivicsPR.16.a).
TECH.9.4.12.DC.2	Compare and contrast international differences in copyright laws and ethics.
TECH.9.4.12.DC.3	Evaluate the social and economic implications of privacy in the context of safety, law, or ethics (e.g., 6.3.12.HistoryCA.1).
TECH.9.4.12.DC.5	Debate laws and regulations that impact the development and use of software.
TECH.9.4.12.DC.7	Evaluate the influence of digital communities on the nature, content and responsibilities of careers, and other aspects of society (e.g., 6.1.12.CivicsPD.16.a).
TECH.9.4.12.DC.8	Explain how increased network connectivity and computing capabilities of everyday objects allow for innovative technological approaches to climate protection.
TECH.9.4.12.TL.3	Analyze the effectiveness of the process and quality of collaborative environments.

TECH.9.4.12.IML.1	Compare search browsers and recognize features that allow for filtering of information.
TECH.9.4.12.IML.2	Evaluate digital sources for timeliness, accuracy, perspective, credibility of the source, and relevance of information, in media, data, or other resources (e.g., NJSLSA.W8, Social Studies Practice: Gathering and Evaluating Sources.

NJ Computer Science Standards

CS.9-12.8.1.12.CS.1	Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
CS.9-12.8.1.12.CS.2	Model interactions between application software, system software, and hardware.
CS.9-12.8.1.12.CS.3	Compare the functions of application software, system software, and hardware.
CS.9-12.8.1.12.CS.4	Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
CS.9-12.8.1.12.IC.1	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
CS.9-12.8.1.12.IC.3	Predict the potential impacts and implications of emerging technologies on larger social, economic, and political structures, using evidence from credible sources.
CS.9-12.8.1.12.NI.1	Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.
CS.9-12.8.1.12.NI.2	Evaluate security measures to address various common security threats.
CS.9-12.8.1.12.NI.3	Explain how the needs of users and the sensitivity of data determine the level of security implemented.
CS.9-12.8.1.12.NI.4	Explain how decisions on methods to protect data are influenced by whether the data is at rest, in transit, or in use.
CS.9-12.8.2.12.EC.1	Analyze controversial technological issues and determine the degree to which individuals, businesses, and governments have an ethical role in decisions that are made.
CS.9-12.8.2.12.EC.2	Assess the positive and negative impacts of emerging technologies on developing countries and evaluate how individuals, non-profit organizations, and governments have responded.
CS.9-12.8.2.12.NT.1	Explain how different groups can contribute to the overall design of a product.
CS.9-12.8.2.12.ETW.1	Evaluate ethical considerations regarding the sustainability of environmental resources that are used for the design, creation, and maintenance of a chosen product.
CS.9-12.8.2.12.ITH.1	Analyze a product to determine the impact that economic, political, social, and/or cultural factors have had on its design, including its design constraints.
CS.9-12.8.2.12.ITH.3	Analyze the impact that globalization, social media, and access to open source technologies has had on innovation and on a society's economy, politics, and culture.

Transfer Goals and Career Ready Practices

Transfer Goals

Students will be able to independently use their learning to explore the technical challenges and questions that arise from the need to represent digital informations in computers and transfer it between people and computational devices.

Concepts

Essential Questions

- Why are long text messages sometime delivered out of order?
- Are innovators responsible for the harmful effects of their computing innovations, even if those effects were unintentional? Why or why not?
- What app or computer software do you use most often and would have a hard time going without? How does this software solve a problem for you or benefit you?
- When an Internet service outage occurs in a different part of your town or city, how are you still able to access the Internet?

Understandings

Students will understand that...

- CSN-1 Computer systems and networks facilitate the transfer of data.
- IOC-1 While computing innovations are typically designed to achieve a specific purpose, they may have unintended consequences.

Critical Knowledge and Skills

Knowledge

Students will know:

- CSN-1.A.1 A computing device is a physical artifact that can run a program. Some examples include computers, tablets, servers, routers, and smart sensors.
- CSN-1.A.2 A computing system is a group of computing devices and programs working together for a common purpose.
- CSN-1.A.3 A computer network is a group of interconnected computing devices capable of sending or receiving data.
- CSN-1.A.4 A computer network is a type of computing system.

- CSN-1.A.5 A path between two computing devices on a computer network (a sender and a receiver) is a sequence of directly connected computing devices that begins at the sender and ends at the receiver.
- CSN-1.A.6 Routing is the process of finding a path from sender to receiver.
- CSN-1.A.7 The bandwidth of a computer network is the maximum amount of data that can be sent in a fixed amount of time.
- CSN-1.A.8 Bandwidth is usually measured in bits per second.
- CSN-1.B.1 The Internet is a computer network consisting of interconnected networks that use standardized, open (nonproprietary) communication protocols.
- CSN-1.B.2 Access to the Internet depends on the ability to connect a computing device to an Internetconnected device.
- CSN-1.B.3 A protocol is an agreed-upon set of rules that specify the behavior of a system.
- CSN-1.B.4 The protocols used in the Internet are open, which allows users to easily connect additional computing devices to the Internet.
- CSN-1.B.5 Routing on the Internet is usually dynamic; it is not specified in advance.
- CSN-1.B.6 The scalability of a system is the capacity for the system to change in size and scale to meet new demands.
- CSN-1.B.7 The Internet was designed to be scalable.
- CSN-1.C.1 Information is passed through the Internet as a data stream. Data streams contain chunks of data, which are encapsulated in packets.
- CSN-1.C.2 Packets contain a chunk of data and metadata used for routing the packet between the origin and the destination on the Internet, as well as for data reassembly.
- CSN-1.C.3 Packets may arrive at the destination in order, out of order, or not at all.
- CSN-1.C.4 IP, TCP, and UDP are common protocols used on the Internet.
- CSN-1.D.1 The World Wide Web is a system of linked pages, programs, and files.
- CSN-1.D.2 HTTP is a protocol used by the World Wide Web.
- CSN-1.D.3 The World Wide Web uses the Internet.
- CSN-1.E.1 The Internet has been engineered to be fault tolerant, with abstractions for routing and transmitting data.
- CSN-1.E.2 Redundancy is the inclusion of extra components that can be used to mitigate failure of a system if other components fail.
- CSN-1.E.3 One way to accomplish network redundancy is by having more than one path between any two connected devices.
- CSN-1.E.4 If a particular device or connection on the Internet fails, subsequent data will be sent via a different route, if possible.
- CSN-1.E.5 When a system can support failures and still continue to function, it is called fault-tolerant. This is important because elements of complex systems fail at unexpected times, often in groups, and fault tolerance allows users to continue to use the network.
- CSN-1.E.6 Redundancy within a system often requires additional resources but can provide the benefit of fault tolerance.
- CSN-1.E.7 The redundancy of routing options between two points increases the reliability of the Internet and helps it scale to more devices and more people.
- IOC-1.C.1 Internet access varies between socioeconomic, geographic, and demographic characteristics, as well as between countries.
- IOC-1.C.2 The “digital divide” refers to differing access to computing devices and the Internet, based on socioeconomic, geographic, or demographic characteristics.

- IOC-1.C.3 The digital divide can affect both groups and individuals.
- IOC-1.C.4 The digital divide raises issues of equity, access, and influence, both globally and locally
- IOC-1.C.5 The digital divide is affected by the actions of individuals, organizations, and governments.

Skills

Students will be able to:

- CSN-1.A Explain how computing devices work together in a network.
- CSN-1.B Explain how the Internet works.
- CSN-1.C Explain how data are sent through the Internet via packets.
- CSN-1.D Describe the differences between the Internet and the World Wide Web.
- CSN-1.E For fault-tolerant systems, like the Internet: a. Describe the benefits of fault tolerance. b. Explain how a given system is fault-tolerant. c. Identify vulnerabilities to failure in a system.
- IOC-1.C Describe issues that contribute to the digital divide

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- Code Studio Review and Reflection Questions
- Code.org Activity Guides
- Internet Simulator Activities

School Summative Assessment Pan

- Project - Internet Dilemmas
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/s/csp2-2022)

Supplementary Resources

- Blown to Bits (http://www.bitsbook.com/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace. These tools
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.

- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

- Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

- Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.
- All assignments have been created in the student's native language.
- Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

- Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

- Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.
- All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.
- All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

LANGUAGE ARTS:

- Students will be expected to read and synthesize technical documents.
- Students will use the correct vocabulary in context to explain the function of the Internet and digital representation of data.

SCIENCE:

- Students will create hypotheses, then build and simulate systems to test these hypotheses.

SOCIAL STUDIES:

- Students will discuss the global effects of computing (both beneficial and harmful) on people and society.
- Students will explore the history of the development of the Internet.

WORLD LANGUAGES:

- Students will explore possible differences in Internet experience due to language and culture.

VISUAL/PERFORMING ARTS:

- Students will present a one pager and part of their unit project.

BUSINESS EDUCATION:

- Students will explore the effects of the Internet on the conducting of business at a global scale.

GLOBAL AWARENESS:

- Students will discuss the global effects of computing (both beneficial and harmful) on people and society.

Learning Plan / Pacing Guide

WEEK 1:

- Lesson 1: Welcome to the Internet
- Lesson 2: Building a Network
- Lesson 3: The Need for Addressing
- Lesson 4: Routers and Redundancy
- Lesson 5: Packets

WEEK 2:

- Lesson 6: HTTP and DNS
- Lessons 7-8: Project - Internet Dilemmas
- Lesson 9: Unit Assessment

Unit 3: Introduction to App Design (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 1**
Length:	**2.5 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

• BIG IDEA 1: CREATIVE DEVELOPMENT (CRD) When developing computing innovations, developers can use a formal, iterative design process or a less rigid process of experimentation. While using either approach, developers will encounter phases of investigating and reflecting, designing, prototyping, and testing. Additionally, collaboration is an important tool at any phase of development, because considering multiple perspectives allows for improvement of innovations.

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

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

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CI.3	Investigate new challenges and opportunities for personal growth, advancement, and transition (e.g., 2.1.12.PGD.1).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.
CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
CS.9-12.8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
CS.9-12.8.1.12.AP.6	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
CS.9-12.8.1.12.AP.7	Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
CS.9-12.8.1.12.AP.8	Evaluate and refine computational artifacts to make them more usable and accessible.
CS.9-12.8.1.12.AP.9	Collaboratively document and present design decisions in the development of complex programs.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to use the JavaScript programming language to create small applications that live on the web.

Concepts

Essential Questions

- How has working collaboratively with other students improved an overall project?
- What apps or programs have you stopped using because you didn't like the design of how you interacted with it?
- What are some ways you can collect additional feedback on your program to use for improvements?

- What are some ways you currently plan your work before starting a project?

Understandings

- CRD-1 Incorporating multiple perspectives through collaboration improves computing innovations as they are developed.
- CRD-2 Developers create and innovate using an iterative design process that is userfocused, that incorporates implementation/feedback cycles, and that leaves ample room for experimentation and risk-taking.

Critical Knowledge and Skills

Knowledge

Students will know:

- CRD-1.A.1 A computing innovation includes a program as an integral part of its function.
- CRD-1.A.2 A computing innovation can be physical (e.g., self-driving car), nonphysical computing software (e.g., picture editing software), or a nonphysical computing concept (e.g., e-commerce).
- CRD-1.A.3 Effective collaboration produces a computing innovation that reflects the diversity of talents and perspectives of those who designed it.
- CRD-1.A.4 Collaboration that includes diverse perspectives helps avoid bias in the development of computing innovations.
- CRD-1.A.5 Consultation and communication with users are important aspects of the development of computing innovations.
- CRD-1.A.6 Information gathered from potential users can be used to understand the purpose of a program from diverse perspectives and to develop a program that fully incorporates these perspectives.
- CRD-1.B.1 Online tools support collaboration by allowing programmers to share and provide feedback on ideas and documents.
- CRD-1.B.2 Common models such as pair programming exist to facilitate collaboration.
- CRD-1.C.1 Effective collaborative teams practice interpersonal skills, including but not limited to: § communication § consensus building § conflict resolution § negotiation
- CRD-2.A.1 The purpose of computing innovations is to solve problems or to pursue interests through creative expression.
- CRD-2.A.2 An understanding of the purpose of a computing innovation provides developers with an improved ability to develop that computing innovation.
- CRD-2.B.1 A program is a collection of program statements that performs a specific task when run by a computer. A program is often referred to as software
- CRD-2.B.2 A code segment is a collection of program statements that is part of a program.
- CRD-2.B.3 A program needs to work for a variety of inputs and situations.

- CRD-2.B.4 The behavior of a program is how a program functions during execution and is often described by how a user interacts with it.
- CRD-2.B.5 A program can be described broadly by what it does, or in more detail by both what the program does and how the program statements accomplish this function.
- CRD-2.C.1 Program inputs are data sent to a computer for processing by a program. Input can come in a variety of forms, such as tactile, audio, visual, or text.
- CRD-2.C.2 An event is associated with an action and supplies input data to a program.
- CRD-2.C.3 Events can be generated when a key is pressed, a mouse is clicked, a program is started, or any other defined action occurs that affects the flow of execution.
- CRD-2.C.4 Inputs usually affect the output produced by a program
- CRD-2.C.5 In event-driven programming, program statements are executed when triggered rather than through the sequential flow of control.
- CRD-2.C.6 Input can come from a user or other programs.
- CRD-2.D.1 Program outputs are any data sent from a program to a device. Program output can come in a variety of forms, such as tactile, audio, visual, or text.
- CRD-2.D.2 Program output is usually based on a program's input or prior state (e.g., internal values).
- CRD-2.E.1 A development process can be ordered and intentional, or exploratory in nature.
- CRD-2.E.2 There are multiple development processes. The following phases are commonly used when developing a program: § investigating and reflecting § designing § prototyping § testing
- CRD-2.E.3 A development process that is iterative requires refinement and revision based on feedback, testing, or reflection throughout the process. This may require revisiting earlier phases of the process.
- CRD-2.E.4 A development process that is incremental is one that breaks the problem into smaller pieces and makes sure each piece works before adding it to the whole.
- CRD-2.F.1 The design of a program incorporates investigation to determine its requirements.
- CRD-2.F.2 Investigation in a development process is useful for understanding and identifying the program constraints, as well as the concerns and interests of the people who will use the program.
- CRD-2.F.3 Some ways investigation can be performed are as follows: § collecting data through surveys § user testing § interviews § direct observations
- CRD-2.F.4 Program requirements describe how a program functions and may include a description of user interactions that a program must provide.
- CRD-2.F.5 A program's specification defines the requirements for the program.
- CRD-2.F.6 In a development process, the design phase outlines how to accomplish a given program specification.
- CRD-2.F.7 The design phase of a program may include: § § planning and storyboarding § organizing the program into modules and functional components § creation of diagrams that represent the layouts of the user interface § development of a testing strategy for the program
- CRD-2.G.1 Program documentation is a written description of the function of a code segment, event, procedure, or program and how it was developed.
- CRD-2.G.2 Comments are a form of program documentation written into the program to be read by people and do not affect how a program runs.
- CRD-2.G.3 Programmers should document a program throughout its development.
- CRD-2.G.4 Program documentation helps in developing and maintaining correct programs when working individually or in collaborative programming environments.
- CRD-2.G.5 Not all programming environments support comments, so other methods of documentation may be required.

- CRD-2.H.1 It is important to acknowledge any code segments that were developed collaboratively or by another source.
- CRD-2.H.2 Acknowledgement of a code segment(s) written by someone else and used in a program can be in the program documentation. The acknowledgement should include the origin or original author's name.

Skills

Students will be able to:

- CRD-1.A Explain how computing innovations are improved through collaboration.
- CRD-1.B Explain how computing innovations are developed by groups of people.
- CRD-1.C Demonstrate effective interpersonal skills during collaboration.
- CRD-2.A Describe the purpose of a computing innovation.
- CRD-2.B Explain how a program or code segment functions.
- CRD-2.C Identify input(s) to a program
- CRD-2.D Identify output(s) produced by a program.
- CRD-2.E Develop a program using a development process
- CRD-2.F Design a program and its user interface.
- CRD-2.G Describe the purpose of a code segment or program by writing documentation.
- CRD-2.H Acknowledge code segments used from other sources.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- App Lab Activities
- Code Studio Review and Reflection Questions
- Code.org Activity Guides

School Summative Assessment Pan

- Project - Designing an App
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/s/csp3-2022)

Supplementary Resources

- AppLab Documentation (https://docs.code.org/applab/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Code.org App Lab**
  - The App Lab provides a IDE in which students can program using a limited version of JavaScript, either via text or drag and drop.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❑ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❑ All assignments have been created in the student's native language.

❑ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer

graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lesson 1: Introduction to Apps
- Lesson 2: Introduction to Design Mode
- Lesson 3: Project - Designing an App Part 1
- Lesson 4: Project - Designing an App Part 2

WEEK 2:

- Lesson 5: The Need for Programming Languages
- Lesson 6: Intro to Programming
- Lesson 7: Debugging
- Lesson 8: Project - Designing an App Part 3
- Lesson 9: Project - Designing an App Part 4

WEEK 3:

- Lesson 10: Project - Designing an App Part 5
- Lesson 11: Unit Assessment

Unit 4: Variables, Conditionals, and Functions (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 2**
Length:	**3 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

• BIG IDEA 1: CREATIVE DEVELOPMENT (CRD) When developing computing innovations, developers can use a formal, iterative design process or a less rigid process of experimentation. While using either approach, developers will encounter phases of investigating and reflecting, designing, prototyping, and testing. Additionally, collaboration is an important tool at any phase of development, because considering multiple perspectives allows for improvement of innovations.

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

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CI.3	Investigate new challenges and opportunities for personal growth, advancement, and transition (e.g., 2.1.12.PGD.1).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.

CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
CS.9-12.8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
CS.9-12.8.1.12.AP.6	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
CS.9-12.8.1.12.AP.7	Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
CS.9-12.8.1.12.AP.8	Evaluate and refine computational artifacts to make them more usable and accessible.
CS.9-12.8.1.12.AP.9	Collaboratively document and present design decisions in the development of complex programs.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to use the JavaScript programming language to use variables, conditionals, and functions to small applications that live on the web.

Concepts

Essential Questions

- How can we store data in a program to solve problems?
- How do video games group the different actions for a player based on what key is pressed on the keyboard or controller? How do apps group different actions together based on user interaction, such as pressing buttons?
- What apps or programs have you stopped using because you didn't like the design of how you interacted with it?
- What are some ways you currently plan your work before starting a project?
- What might happen if you completed the steps in your regular morning routine to get ready and go to school in a different order? How might the reordering affect the decisions you make each morning?

Understandings

- AAP-1 To find specific solutions to generalizable problems, programmers represent and organize data in multiple ways.
- AAP-2 The way statements are sequenced and combined in a program determines the computed result. Programs incorporate iteration and selection constructs to represent repetition and make decisions to handle varied input values.
- AAP-3 Programmers break down problems into smaller and more manageable pieces. By creating procedures and leveraging parameters, programmers generalize processes that can be reused. Procedures allow programmers to draw upon existing code that has already been tested, allowing them to write programs more quickly and with more confidence.
- CRD-2 Developers create and innovate using an iterative design process that is userfocused, that incorporates implementation/feedback cycles, and that leaves ample room for experimentation and risk-taking.

Critical Knowledge and Skills

Knowledge

Students will know:

- CRD-2.I.1 A logic error is a mistake in the algorithm or program that causes it to behave incorrectly or unexpectedly.
- AAP-1.A.1 A variable is an abstraction inside a program that can hold a value. Each variable has associated data storage that represents one value at a time, but that value can be a list or other collection that in turn contains multiple values.
- AAP-1.A.2 Using meaningful variable names helps with the readability of program code and understanding of what values are represented by the variables.
- AAP-1.A.3 Some programming languages provide types to represent data, which are referenced using variables. These types include numbers, Booleans, lists, and strings.
- AAP-1.A.4 Some values are better suited to representation using one type of datum rather than another.
- AAP-1.B.1 The assignment operator allows a program to change the value represented by a variable.
- AAP-1.B.2 The exam reference sheet provides the “<-- ” operator to use for assignment.
- AAP-1.B.3 The value stored in a variable will be the most recent value assigned.
- AAP-2.A.1 An algorithm is a finite set of instructions that accomplish a specific task.
- AAP-2.A.2 Beyond visual and textual programming languages, algorithms can be expressed in a variety of ways, such as natural language, diagrams, and pseudocode.
- AAP-2.A.3 Algorithms executed by programs are implemented using programming languages.
- AAP-2.A.4 Every algorithm can be constructed using combinations of sequencing, selection, and iteration.
- AAP-2.B.1 Sequencing is the application of each step of an algorithm in the order in which the code

statements are given

- AAP-2.B.2 A code statement is a part of program code that expresses an action to be carried out.
- AAP-2.B.3 An expression can consist of a value, a variable, an operator, or a procedure call that returns a value
- AAP-2.B.4 Expressions are evaluated to produce a single value.
- AAP-2.B.5 The evaluation of expressions follows a set order of operations defined by the programming language.
- AAP-2.B.6 Sequential statements execute in the order they appear in the code segment.
- AAP-2.B.7 Clarity and readability are important considerations when expressing an algorithm in a programming language.
- AAP-2.C.1 Arithmetic operators are part of most programming languages and include addition, subtraction, multiplication, division, and modulus operators.
- AAP-2.C.2 The exam reference sheet provides a MOD b, which evaluates to the remainder when a is divided by b. Assume that a is an integer greater than or equal to 0 and b is an integer greater than 0. F
- AAP-2.C.3 The exam reference sheet provides the arithmetic operators +, -, *, /, and MOD. Text and Block: § a + b § a – b § a * b § a / b § a MOD b These are used to perform arithmetic on a and b. For example, 17 / 5 evaluates to 3.4.
- AAP-2.C.4 The order of operations used in mathematics applies when evaluating expressions. The MOD operator has the same precedence as the * and / operators.
- AAP-2.E.1 A Boolean value is either true or false.
- AAP-2.E.2 The exam reference sheet provides the following relational operators: =, ≠, >, <, ≥, and ≤. Text and Block: § a = b § a ≠ b § a > b § a < b § a ≥ b § a ≤ b These are used to test the relationship between two variables, expressions, or values. A comparison using a relational operator evaluates to a Boolean value. For example, a = b evaluates to true if a and b are equal; otherwise, it evaluates to false.
- AAP-2.F.1 The exam reference sheet provides the logical operators NOT, AND, and OR, which evaluate to a Boolean value
- AAP-2.F.2 The exam reference sheet provides Text: NOT condition Block: NOT condition which evaluates to true if condition is false; otherwise it evaluates to false.
- AAP-2.F.3 The exam reference sheet provides Text: condition1 AND condition2 Block: condition1 AND condition2 which evaluates to true if both condition1 and condition2 are true; otherwise it evaluates to false.
- AAP-2.F.4 The exam reference sheet provides Text: condition1 OR condition2 Block: condition1 OR condition2 which evaluates to true if condition1 is true or if condition2 is true or if both condition1 and condition2 are true; otherwise it evaluates to false.
- AAP-2.F.5 The operand for a logical operator is either a Boolean expression or a single Boolean value.
- AAP-2.G.1 Selection determines which parts of an algorithm are executed based on a condition being true or false.
- AAP-2.H.1 Conditional statements, or “if-statements,” affect the sequential flow of control by executing different statements based on the value of a Boolean expression.
- AAP-2.H.2 The exam reference sheet provides Text: IF(condition) { <block of statements> } Block: IF condition block of statements in which the code in block of statements is executed if the Boolean expression condition evaluates to true; no action is taken if condition evaluates to false.
- AAP-2.H.3 The exam reference sheet provides Text: IF(condition) { <first block of statements> } ELSE { <second block of statements> } Block: IF condition first block of statements ELSE second block of statements in which the code in first block of statements is executed if the Boolean expression condition evaluates to true; otherwise, the code in second block of statements is executed.
- AAP-2.I.1 Nested conditional statements consist of conditional statements within conditional statements.

- AAP-3.E.1 The exam reference sheet provides Text: RANDOM(a, b) Block: RANDOM a, b which generates and returns a random integer from a to b, inclusive. Each result is equally likely to occur. For example, RANDOM(1, 3) could return 1, 2, or 3.
- AAP-3.E.2 Using random number generation in a program means each execution may produce a different result.
- CRD-2.I.2 A syntax error is a mistake in the program where the rules of the programming language are not followed.
- CRD-2.I.3 A run-time error is a mistake in the program that occurs during the execution of a program. Programming languages define their own runtime errors
- CRD-2.I.4 An overflow error is an error that occurs when a computer attempts to handle a number that is outside of the defined range of values.
- CRD-2.I.5 The following are effective ways to find and correct errors: § test cases § hand tracing § visualizations § debuggers § adding extra output statement(s)
- CRD-2.J.1 In the development process, testing uses defined inputs to ensure that an algorithm or program is producing the expected outcomes. Programmers use the results from testing to revise their algorithms or programs.
- CRD-2.J.2 Defined inputs used to test a program should demonstrate the different expected outcomes that are at or just beyond the extremes (minimum and maximum) of input data.
- CRD-2.J.3 Program requirements are needed to identify appropriate defined inputs for testing.

Skills

Students will be able to:

- AAP-1.A Represent a value with a variable.
- AAP-1.B Determine the value of a variable as a result of an assignment. 4
- AAP-2.A Express an algorithm that uses sequencing without using a programming language.
- AAP-2.B Represent a step-by-step algorithmic process using sequential code statements.
- AAP-2.C Evaluate expressions that use arithmetic operators.
- AAP-2.E For relationships between two variables, expressions, or values, write and evaluate expressions using relational operators.
- AAP-2.G Express an algorithm that uses selection without using a programming language.
- AAP-2.H For selection, write and determine the result of conditional statements.
- AAP-2.I For nested selection, write and determine the result of nested conditional statements.
- AAP-3.E For generating random values: a. write expressions to generate possible values. b. evaluate expressions to determine the possible results.
- CRD-2.I For errors in an algorithm or program: a. Identify the error b. Correct the error.
- CRD-2.J Identify inputs and corresponding expected outputs or behaviors that can be used to check the correctness of an algorithm or program.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- App Lab Activities
- Code Studio Review and Reflection Questions
- Code.org Activity Guides

School Summative Assessment Pan

- Project: Decision Maker App
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/s/csp4-2022/)

Supplementary Resources

- AppLab Documentation (https://docs.code.org/applab/)
- Blown to Bits (http://bitsbook.com)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Code.org App Lab**
  - The App Lab provides a IDE in which students can program using a limited version of JavaScript, either via text or drag and drop.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and

see results upon completion of the assignments to allow for 21st century learning.

- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❑ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❑ All assignments have been created in the student's native language.

❑ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lessons 1-4: Variables Explore, Investigate, Practice, Make
- Lesson 5: Conditionals Explore

WEEK 2:

- Lesson 6-8: Conditionals Investigate, Practice, Make
- Lessons 9-10: Functions Explore/Investigate, Practice

WEEK 3:

- Lesson 11: Traversals Make
- Lessons 12-14: Decision Maker App Parts 1-3

- Lesson 15: Unit Assessment

Unit 5: Lists, Loops, and Traversals (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 2**
Length:	**4 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA 1: CREATIVE DEVELOPMENT (CRD) When developing computing innovations, developers can use a formal, iterative design process or a less rigid process of experimentation. While using either approach, developers will encounter phases of investigating and reflecting, designing, prototyping, and testing. Additionally, collaboration is an important tool at any phase of development, because considering multiple perspectives allows for improvement of innovations.
- BIG IDEA 3: ALGORITHMS AND PROGRAMMING (AAP) Programmers integrate algorithms and abstraction to create programs for creative purposes and to solve problems. Using multiple program statements in a specified order, making decisions, and repeating the same process multiple times are the building blocks of programs. Incorporating elements of abstraction—by breaking problems down into interacting pieces, each with their own purpose—makes writing complex programs easier. Programmers need to think algorithmically and use abstraction to define and interpret processes that are used in a program.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CI.3	Investigate new challenges and opportunities for personal growth, advancement, and transition (e.g., 2.1.12.PGD.1).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.

CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
CS.9-12.8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
CS.9-12.8.1.12.AP.6	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
CS.9-12.8.1.12.AP.7	Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
CS.9-12.8.1.12.AP.8	Evaluate and refine computational artifacts to make them more usable and accessible.
CS.9-12.8.1.12.AP.9	Collaboratively document and present design decisions in the development of complex programs.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to use the JavaScript programming language to use lists, loops, and traversals to write small applications that live on the web.

Concepts

Essential Questions

- How can we store data in a program to solve problems?
- How do video games group the different actions for a player based on what key is pressed on the keyboard or controller? How do apps group different actions together based on user interaction, such as pressing buttons?
- What might happen if you completed the steps in your regular morning routine to get ready and go to school in a different order? How might the reordering affect the decisions you make each morning?

Understandings

- AAP-1 To find specific solutions to generalizable problems, programmers represent and organize data in multiple ways.

- AAP-2 The way statements are sequenced and combined in a program determines the computed result. Programs incorporate iteration and selection constructs to represent repetition and make decisions to handle varied input values.
- AAP-3 Programmers break down problems into smaller and more manageable pieces. By creating procedures and leveraging parameters, programmers generalize processes that can be reused. Procedures allow programmers to draw upon existing code that has already been tested, allowing them to write programs more quickly and with more confidence.

Critical Knowledge and Skills

Knowledge

Students will know:

- AAP-1.C.1 A list is an ordered sequence of elements. For example, [value1, value2, value3, ...] describes a list where value1 is the first element, value2 is the second element, value3 is the third element, and so on
- AAP-1.C.2 An element is an individual value in a list that is assigned a unique index.
- AAP-1.C.3 An index is a common method for referencing the elements in a list or string using natural numbers.
- AAP-1.C.4 A string is an ordered sequence of characters.
- AAP-1.D.1 Data abstraction provides a separation between the abstract properties of a data type and the concrete details of its representation.
- AAP-1.D.2 Data abstractions manage complexity in programs by giving a collection of data a name without referencing the specific details of the representation.
- AAP-1.D.3 Data abstractions can be created using lists
- AAP-1.D.4 Developing a data abstraction to implement in a program can result in a program that is easier to develop and maintain.
- AAP-1.D.5 Data abstractions often contain different types of elements.
- AAP-1.D.6 The use of lists allows multiple related items to be treated as a single value. Lists are referred to by different names, such as array, depending on the programming language.
- AAP-1.D.7 The exam reference sheet provides the notation [value1, value2, value3, ...] to create a list with those values as the first, second, third, and so on items.
- AAP-2.D.1 String concatenation joins together two or more strings end-to-end to make a new string.
- AAP-2.D.2 A substring is part of an existing string
- AAP-2.J.1 Iteration is a repeating portion of an algorithm. Iteration repeats a specified number of times or until a given condition is met.
- AAP-2.K.2 The exam reference sheet provides Text: REPEAT n TIMES { <block of statements> } Block: REPEAT n TIMES block of statements in which the block of statements is executed n times.
- AAP-2.K.3 The exam reference sheet provides Text: REPEAT UNTIL(condition) { <block of statements>

} Block: REPEAT UNTIL block of statements condition in which the code in block of statements is repeated until the Boolean expression condition evaluates to true.

- AAP-2.K.4 In REPEAT UNTIL(condition) iteration, an infinite loop occurs when the ending condition will never evaluate to true.

- AAP-2.K.5 In REPEAT UNTIL(condition) iteration, if the conditional evaluates to true initially, the loop body is not executed at all, due to the condition being checked before the loop.

- AAP-2.N.1 The exam reference sheet provides basic operations on lists, including: § accessing an element by index Text: aList[i] Block: aList i accesses the element of aList at index i. The first element of aList is at index 1 and is accessed using the notation aList[1]. § assigning a value of an element of a list to a variable Text: x aList [i] Block: x aList i assigns the value of aList[i] to the variable x. § assigning a value to an element of a list Text: aList[i] x Block: aList i x assigns the value of x to aList[i]Text: aList[i] aList[j] Block: aList i j aList assigns the value of aList[j] to aList[i]. § inserting elements at a given index Text: INSERT(aList, i, value) Block: INSERT aList, i, value shifts to the right any values in aList at indices greater than or equal to i. The length of the list is increased by 1, and value is placed at index i in aList. § adding elements to the end of the list Text: APPEND(aList, value) Block: APPEND aList, value increases the length of aList by 1, and value is placed at the end of aList. § removing elements Text: REMOVE(aList, i) Block: REMOVE aList, i removes the item at index i in aList and shifts to the left any values at indices greater than i. The length of aList is decreased by 1. § determining the length of a list Text: LENGTH(aList) Block: LENGTH aList evaluates to the number of elements currently in aList.

- AAP-2.N.2 List procedures are implemented in accordance with the syntax rules of the programming language.

- AAP-2.O.1 Traversing a list can be a complete traversal, where all elements in the list are accessed, or a partial traversal, where only a portion of elements are accessed.

- AAP-2.O.2 Iteration statements can be used to traverse a list.

- AAP-2.O.3 The exam reference sheet provides Text: FOR EACH item IN aList { <block of statements> } Block: FOR EACH item IN aList block of statements The variable item is assigned the value of each element of aList sequentially, in order, from the first element to the last element. The code in block of statements is executed once for each assignment of item.

- AAP-2.O.4 Knowledge of existing algorithms that use iteration can help in constructing new algorithms. Some examples of existing algorithms that are often used with lists include: § determining a minimum or maximum value in a list § computing a sum or average of a list of numbers

- AAP-2.O.5 Linear search or sequential search algorithms check each element of a list, in order, until the desired value is found or all elements in the list have been checked.

- AAP-3.F.1 Simulations are abstractions of more complex objects or phenomena for a specific purpose.

- AAP-3.F.2 A simulation is a representation that uses varying sets of values to reflect the changing state of a phenomenon.

- AAP-3.F.3 Simulations often mimic real-world events with the purpose of drawing inferences, allowing investigation of a phenomenon without the constraints of the real world.

- AAP-3.F.4 The process of developing an abstract simulation involves removing specific details or simplifying functionality.

- AAP-3.F.5 Simulations can contain bias derived from the choices of real-world elements that were included or excluded.

- AAP-3.F.6 Simulations are most useful when real-world events are impractical for experiments (e.g., too big, too small, too fast, too slow, too expensive, or too dangerous).

- AAP-3.F.7 Simulations facilitate the formulation and refinement of hypotheses related to the objects or phenomena under consideration.

- AAP-3.F.8 Random number generators can be used to simulate the variability that exists in the real world.

Skills

Students will be able to:

- AAP-1.C Represent a list or string using a variable
- AAP-1.D For data abstraction: a. Develop data abstraction using lists to store multiple elements. b. Explain how the use of data abstraction manages complexity in program code.
- AAP-2.D Evaluate expressions that manipulate strings
- AAP-2.J Express an algorithm that uses iteration without using a programming language.
- AAP-2.K For iteration: a. Write iteration statements. b. Determine the result or side effect of iteration statements.
- AAP-2.N For list operations: a. Write expressions that use list indexing and list procedures. b. Evaluate expressions that use list indexing and list procedures.
- AAP-2.O For algorithms involving elements of a list: a. Write iteration statements to traverse a list. b. Determine the result of an algorithm that includes list traversals.
- AAP-3.F For simulations: a. Explain how computers can be used to represent real-world phenomena or outcomes. b. Compare simulations with real-world contexts.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- App Lab Activities
- Code Studio Review and Reflection Questions
- Code.org Activity Guides

School Summative Assessment Pan

- Project Hackathon
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/s/csp5-2022/)

Supplementary Resources

- AppLab Documentation (https://docs.code.org/applab/)
- Blown to Bits (http://bitsbook.com)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Code.org App Lab**
  - The App Lab provides a IDE in which students can program using a limited version of JavaScript, either via text or drag and drop.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❑ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❑ All assignments have been created in the student’s native language.

❑ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.

- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lessons 1-4: Lists Explore, Investigate, Practice, Make
- Lesson 5: Loops Explore

WEEK 2:

- Lesson 6-8: Loops Investigate, Practice, Make
- Lessons 9-10: Traversals Explore, Investigate

WEEK 3:

- Lessons 11-12: Traversals Practice, Make
- Lessons 13-15: Project Hackathon Parts 1-3

WEEK 4:

- Lessons 16-17: Project Hackathon Parts 4-5
- Lesson 18: Unit Assessment

Unit 6: Algorithms (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 3**
Length:	**2 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA 3: ALGORITHMS AND PROGRAMMING (AAP) Programmers integrate algorithms and abstraction to create programs for creative purposes and to solve problems. Using multiple program statements in a specified order, making decisions, and repeating the same process multiple times are the building blocks of programs. Incorporating elements of abstraction—by breaking problems down into interacting pieces, each with their own purpose—makes writing complex programs easier. Programmers need to think algorithmically and use abstraction to define and interpret processes that are used in a program.
- BIG IDEA 4: COMPUTING SYSTEMS AND NETWORKS (CSN) Computer systems and networks are used to transfer data. One of the largest and most commonly used networks is the Internet. Through a series of protocols, the Internet can be used to send and receive information and ideas throughout the world. Transferring and processing information can be slow when done on a single computer, but leveraging multiple computers to do the work at the same time can significantly shorten the time it takes to complete tasks or solve problems.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CI.3	Investigate new challenges and opportunities for personal growth, advancement, and transition (e.g., 2.1.12.PGD.1).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and

	existing algorithms.
CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
CS.9-12.8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
CS.9-12.8.1.12.AP.6	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
CS.9-12.8.1.12.AP.7	Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
CS.9-12.8.1.12.AP.8	Evaluate and refine computational artifacts to make them more usable and accessible.
CS.9-12.8.1.12.AP.9	Collaboratively document and present design decisions in the development of complex programs.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to use the JavaScript programming language to create small applications that live on the web.

Concepts

Essential Questions

- What might happen if you completed the steps in your regular morning routine to get ready and go to school in a different order? How might the reordering affect the decisions you make each morning?
- What types of problems can be solved more easily with a computer, and what types can be solved more easily without a computer? Why?
- What are the benefits of dividing tasks among group members?
- Is there a point where adding another group member would not make completing the task faster? Why?

Understandings

- AAP-2 The way statements are sequenced and combined in a program determines the computed result. Programs incorporate iteration and selection constructs to represent repetition and make decisions to handle varied input values.
- AAP-4 There exist problems that computers cannot solve, and even when a computer can solve a problem, it may not be able to do so in a reasonable amount of time.
- CSN-2 Parallel and distributed computing leverage multiple computers to more quickly solve complex problems or process large data sets.

Critical Knowledge and Skills

Knowledge

Students will know:

- AAP-2.L.1 Algorithms can be written in different ways and still accomplish the same tasks
- AAP-2.L.2 Algorithms that appear similar can yield different side effects or results.
- AAP-2.L.3 Some conditional statements can be written as equivalent Boolean expressions.
- AAP-2.L.4 Some Boolean expressions can be written as equivalent conditional statements.
- AAP-2.L.5 Different algorithms can be developed or used to solve the same problem.
- AAP-2.M.1 Algorithms can be created from an idea, by combining existing algorithms, or by modifying existing algorithms.
- AAP-2.M.2 Knowledge of existing algorithms can help in constructing new ones. Some existing algorithms include: § determining the maximum or minimum value of two or more numbers § computing the sum or average of two or more numbers § identifying if an integer is or is not evenly divisible by another integer § determining a robot's path through a maze
- AAP-2.M.3 Using existing correct algorithms as building blocks for constructing another algorithm has benefits such as reducing development time, reducing testing, and simplifying the identification of errors.
- AAP-2.P.1 The binary search algorithm starts at the middle of a sorted data set of numbers and eliminates half of the data; this process repeat until the desired value is found or all elements have been eliminated.
- AAP-2.P.2 Data must be in sorted order to use the binary search algorithm.
- AAP-2.P.3 Binary search is often more efficient than sequential/linear search when applied to sorted data.
- AAP-4.A.1 A problem is a general description of a task that can (or cannot) be solved algorithmically. An instance of a problem also includes specific input. For example, sorting is a problem; sorting the list (2,3,1,7) is an instance of the problem.
- AAP-4.A.2 A decision problem is a problem with a yes/no answer (e.g., is there a path from A to B?). An optimization problem is a problem with the goal of finding the "best" solution among many (e.g., what is the shortest path from A to B?).
- AAP-4.A.3 Efficiency is an estimation of the amount of computational resources used by an algorithm.

Efficiency is typically expressed as a function of the size of the input.

- AAP-4.A.4 An algorithm's efficiency is determined through formal or mathematical reasoning.
- AAP-4.A.5 An algorithm's efficiency can be informally measured by determining the number of times a statement or group of statements executes.
- AAP-4.A.6 Different correct algorithms for the same problem can have different efficiencies.
- AAP-4.A.7 Algorithms with a polynomial efficiency or slower (constant, linear, square, cube, etc.) are said to run in a reasonable amount of time. Algorithms with exponential or factorial efficiencies are examples of algorithms that run in an unreasonable amount of time.
- AAP-4.A.8 Some problems cannot be solved in a reasonable amount of time because there is no efficient algorithm for solving them. In these cases, approximate solutions are sought.
- AAP-4.A.9 A heuristic is an approach to a problem that produces a solution that is not guaranteed to be optimal but may be used when techniques that are guaranteed to always find an optimal solution are impractical.
- AAP-4.B.1 A decidable problem is a decision problem for which an algorithm can be written to produce a correct output for all inputs (e.g., "Is the number even?").
- AAP-4.B.2 An undecidable problem is one for which no algorithm can be constructed that is always capable of providing a correct yes-or-no answer.
- AAP-4.B.3 An undecidable problem may have some instances that have an algorithmic solution, but there is no algorithmic solution that could solve all instances of the problem.
- CSN-2.A.1 Sequential computing is a computational model in which operations are performed in order one at a time.
- CSN-2.A.2 Parallel computing is a computational model where the program is broken into multiple smaller sequential computing operations, some of which are performed simultaneously.
- CSN-2.A.3 Distributed computing is a computational model in which multiple devices are used to run a program.
- CSN-2.A.4 Comparing efficiency of solutions can be done by comparing the time it takes them to perform the same task.
- CSN-2.A.5 A sequential solution takes as long as the sum of all of its steps.
- CSN-2.A.6 A parallel computing solution takes as long as its sequential tasks plus the longest of its parallel tasks.
- CSN-2.A.7 The "speedup" of a parallel solution is measured in the time it took to complete the task sequentially divided by the time it took to complete the task when done in parallel.
- CSN-2.B.1 Parallel computing consists of a parallel portion and a sequential portion.
- CSN-2.B.2 Solutions that use parallel computing can scale more effectively than solutions that use sequential computing.
- CSN-2.B.3 Distributed computing allows problems to be solved that could not be solved on a single computer because of either the processing time or storage needs involved.
- CSN-2.B.4 Distributed computing allows much larger problems to be solved quicker than they could be solved using a single computer.
- CSN-2.B.5 When increasing the use of parallel computing in a solution, the efficiency of the solution is still limited by the sequential portion. This means that at some point, adding parallel portions will no longer meaningfully increase efficiency.

Skills

Students will be able to:

- AAP-2.L Compare multiple algorithms to determine if they yield the same side effect or result.
- AAP-2.M For algorithms: a. Create algorithms. b. Combine and modify existing algorithms.
- AAP-2.P For binary search algorithms: a. Determine the number of iterations required to find a value in a data set. b. Explain the requirements necessary to complete a binary search.
- AAP-4.A For determining the efficiency of an algorithm: a. Explain the difference between algorithms that run in reasonable time and those that do not. b. Identify solutions where a heuristic solution my be more appropriate.
- AAP-4.B Explain the existence of undecidable problems in computer science.
- CSN-2.A For sequential, parallel, and distributed computing: a. Compare problem solutions. b. Determine the efficiency of solutions.
- CSN-2.B Describe benefits and challenges of parallel and distributed computing.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- App Lab Activities
- Code Studio Review and Reflection Questions
- Code.org Activity Guides

School Summative Assessment Pan

- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://https://studio.code.org/s/csp6-2022)

Supplementary Resources

- AppLab Documentation (https://docs.code.org/applab/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Code.org App Lab**
  - The App Lab provides a IDE in which students can program using a limited version of JavaScript, either via text or drag and drop.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❑ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❑ All assignments have been created in the student's native language.

❑ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of

programming languages, or the impact of computing on historical events and society.

- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

Week 1

- Lesson 1; Algorithms Solve Problems
- Lesson 2: Algorithm Efficiency
- Lesson 3: Unreasonable Time
- Lesson 4: The Limits of Algorithms

Week 2

- Lesson 5: Parallel and Distributed Algorithms
- Lesson 6: Unit Assessment

Unit 7: Parameters, Return, and Libraries (2022)

Content Area:	**Math**
Course(s):	**Generic Course**
Time Period:	**Marking Period 3**
Length:	**2.5 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

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

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CI.3	Investigate new challenges and opportunities for personal growth, advancement, and transition (e.g., 2.1.12.PGD.1).
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.CT.4	Participate in online strategy and planning sessions for course-based, school-based, or other project and determine the strategies that contribute to effective outcomes.
TECH.9.4.12.TL.1	Assess digital tools based on features such as accessibility options, capacities, and utility for accomplishing a specified task (e.g., W.11-12.6.).

NJ Computer Science Standards

CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.
CS.9-12.8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.

CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
CS.9-12.8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
CS.9-12.8.1.12.AP.6	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
CS.9-12.8.1.12.AP.7	Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
CS.9-12.8.1.12.AP.8	Evaluate and refine computational artifacts to make them more usable and accessible.
CS.9-12.8.1.12.AP.9	Collaboratively document and present design decisions in the development of complex programs.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to use the JavaScript programming language to use parameters, return values, and libraries to write small applications that live on the web.

Concepts

Essential Questions

- How do video games group the different actions for a player based on what key is pressed on the keyboard or controller? How do apps group different actions together based on user interaction, such as pressing buttons?

Understandings

- AAP-3 Programmers break down problems into smaller and more manageable pieces. By creating procedures and leveraging parameters, programmers generalize processes that can be reused. Procedures allow programmers to draw upon existing code that has already been tested, allowing them to write programs more quickly and with more confidence.

Critical Knowledge and Skills

Knowledge

Students will know:

- AAP-3.A.1 A procedure is a named group of programming instructions that may have parameters and return values.
- AAP-3.A.7 The exam reference sheet provides the Text: RETURN(expression) Block: RETURN expression statement, which is used to return the flow of control to the point where the procedure was called and to return the value of expression.
- AAP-3.A.2 Procedures are referred to by different names, such as method or function, depending on the programming language.
- AAP-3.A.3 Parameters are input variables of a procedure. Arguments specify the values of the parameters when a procedure is called.
- AAP-3.A.4 A procedure call interrupts the sequential execution of statements, causing the program to execute the statements within the procedure before continuing. Once the last statement in the procedure (or a return statement) has executed, flow of control is returned to the point immediately following where the procedure was called.
- AAP-3.A.5 The exam reference sheet provides procName(arg1, arg2, ...) as a way to call Text: PROCEDURE procName(parameter1, parameter2, ...) { <block of statements> } Block: PROCEDURE procName block of statements parameter1, parameter2,... which takes zero or more arguments; arg1 is assigned to parameter1, arg2 is assigned to parameter2, and so on.
- AAP-3.A.6 The exam reference sheet provides the procedure Text: DISPLAY(expression) Block: DISPLAY expression to display the value of expression, followe by a space.
- AAP-3.A.8 The exam reference sheet provides result procName(arg1, arg2, ...) to assign to result the "value of the procedure" being returned by calling Text: PROCEDURE procName(parameter1, parameter2, ...) { <block of statements> RETURN(expression) } Block: PROCEDURE procName block of statements RETURN expression
- AAP-3.A.9 The exam reference sheet provides procedure Text: INPUT() Block: INPUT which accepts a value from the user and returns the input value.
- AAP-3.B.1 One common type of abstraction is procedural abstraction, which provides a name for a process and allows a procedure to be used only knowing what it does, not how it does it.
- AAP-3.B.2 Procedural abstraction allows a solution to a large problem to be based on the solutions of smaller subproblems. This is accomplished by creating procedures to solve each of the subproblems.
- AAP-3.B.3 The subdivision of a computer program into separate subprograms is called modularity
- AAP-3.B.4 A procedural abstraction may extract shared features to generalize functionality instead of duplicating code. This allows for program code reuse, which helps manage complexity.
- AAP-3.B.5 Using parameters allows procedures to be generalized, enabling the procedures to be reused with a range of input values or arguments.
- AAP-3.B.6 Using procedural abstraction helps improve code readability
- AAP-3.B.7 Using procedural abstraction in a program allows programmers to change the internals of the procedure (to make it faster, more efficient, use less storage, etc.) without needing to notify users of the change as long as what the procedure does is preserved.
- AAP-3.C.1 The exam reference sheet provides Text: PROCEDURE procName(parameter1, parameter2, ...) { <block of statements> } Block: PROCEDURE procName block of statements parameter1, parameter2,... which is used to define a procedure that takes zero or more arguments. The procedure

contains block of statements.

- AAP-3.C.2 The exam reference sheet provides Text: PROCEDURE procName(parameter1, parameter2, ...) { <block of statements> RETURN(expression) } Block: PROCEDURE procName block of statements RETURN expression which is used to define a procedure that takes zero or more arguments. The procedure contains block of statements and returns the value of expression. The RETURN statement may appear at any point inside the procedure and causes an immediate return from the procedure back to the calling statement.
- AAP-3.D.1 A software library contains procedures that may be used in creating new programs.
- AAP-3.D.2 Existing code segments can come from internal or external sources, such as libraries or previously written code.
- AAP-3.D.3 The use of libraries simplifies the task of creating complex programs.
- AAP-3.D.4 Application program interfaces (APIs) are specifications for how the procedures in a library behave and can be used.
- AAP-3.D.5 Documentation for an API/library is necessary in understanding the behaviors provided by the API/library and how to use them.

Skills

Students will be able to:

- AAP-3.A For procedure calls: a. write statements to call procedures. b. Determine the result or effect of a procedure call.
- AAP-3.B Explain how the use of procedural abstraction manages complexity in a program.
- AAP-3.C Develop procedural abstractions to manage complexity in a program by writing procedures.
- AAP-3.D Select appropriate libraries or existing code segments to use in creating new programs.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- App Lab Activities
- Code Studio Review and Reflection Questions
- Code.org Activity Guides

School Summative Assessment Pan

- Project: Make a Library
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://studio.code.org/s/csp7-2022/)

Supplementary Resources

- AppLab Documentation (https://docs.code.org/applab/)
- Blown to Bits (http://bitsbook.com)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Code.org App Lab**
  - The App Lab provides a IDE in which students can program using a limited version of JavaScript, either via text or drag and drop.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❑ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❑ All assignments have been created in the student's native language.

❑ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology,

psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lessons 1-4: Parameters and Return Explore, Investigate, Practice, Make
- Lesson 5: Libraries Explore

WEEK 2:

- Lesson 6-7: Loops Investigate, Practice
- Lessons 8-10: Project: Make a Library Parts 1-3

WEEK 3:

- Lesson 11: Unit Assessment

Unit 8: AP Create Performance Task (2022)

Content Area: **Applied Tech**
Course(s): **Generic Course**
Time Period: **Marking Period 3**
Length: **4 weeks**
Status: **Published**

Content and Directions

This unit is a placeholder for the time that students must be given to complete their AP Create Task.

Directions found here: https://apcentral.collegeboard.org/media/pdf/ap-csp-student-task-directions.pdf

Unit 9: Data (2022)

Content Area:	**Applied Tech**
Course(s):	**Generic Course**
Time Period:	**Marking Period 4**
Length:	**2 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA 2: DATA (DAT) Data are central to computing innovations because they communicate initial conditions to programs and represent new knowledge. Computers consume data, transform data, and produce new data, allowing users to create new information or knowledge to solve problems through the interpretation of those data. Computers store data digitally, which means that the data must be manipulated in order to be presented in a useful way to the user.
- BIG IDEA 5: IMPACT OF COMPUTING (IOC) Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand the potential impacts of our programs and be responsible for the consequences. As computer users, we need to understand any potential beneficial or harmful effects and how to protect ourselves and our privacy when using a computer.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.DC.1	Explain the beneficial and harmful effects that intellectual property laws can have on the creation and sharing of content (e.g., 6.1.12.CivicsPR.16.a).
TECH.9.4.12.DC.3	Evaluate the social and economic implications of privacy in the context of safety, law, or ethics (e.g., 6.3.12.HistoryCA.1).
TECH.9.4.12.DC.4	Explain the privacy concerns related to the collection of data (e.g., cookies) and generation of data through automated processes that may not be evident to users (e.g., 8.1.12.NI.3).
TECH.9.4.12.DC.5	Debate laws and regulations that impact the development and use of software.
TECH.9.4.12.DC.6	Select information to post online that positively impacts personal image and future college and career opportunities.
TECH.9.4.12.TL.4	Collaborate in online learning communities or social networks or virtual worlds to analyze and propose a resolution to a real-world problem (e.g., 7.1.AL.IPERS.6).
TECH.9.4.12.IML.4	Assess and critique the appropriateness and impact of existing data visualizations for an intended audience (e.g., S-ID.B.6b, HS-LS2-4).

NJ Computer Science Standards

CS.9-12.8.1.12.DA.2	Describe the trade-offs in how and where data is organized and stored.
CS.9-12.8.1.12.DA.5	Create data visualizations from large data sets to summarize, communicate, and support different interpretations of real-world phenomena.
CS.9-12.8.1.12.NI.2	Evaluate security measures to address various common security threats.
CS.9-12.8.1.12.NI.3	Explain how the needs of users and the sensitivity of data determine the level of security implemented.
CS.9-12.8.1.12.NI.4	Explain how decisions on methods to protect data are influenced by whether the data is at rest, in transit, or in use.
CS.9-12.8.2.12.EC.1	Analyze controversial technological issues and determine the degree to which individuals, businesses, and governments have an ethical role in decisions that are made.
CS.9-12.8.2.12.EC.3	Synthesize data, analyze trends, and draw conclusions regarding the effect of a technology on the individual, culture, society, and environment and share this information with the appropriate audience.
CS.9-12.8.2.12.ETW.2	Synthesize and analyze data collected to monitor the effects of a technological product or system on the environment.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to construct a well-rounded and balanced view about data in the world around them and to understand the basics of how and why modern encryption works.

Concepts

Essential Questions

- How can you predict the attendance at a school event using data gathered from social media?
- When is it more appropriate to use a computer to analyze data than to complete the analysis by hand?
- Are innovators responsible for the harmful effects of their computing innovations, even if those effects were unintentional? Why or why not?
- What app or computer software do you use most often and would have a hard time going without? How does this software solve a problem for you or benefit you?

Understandings

- IOC-1 While computing innovations are typically designed to achieve a specific purpose, they may have unintended consequences.
- DAT-2 Programs can be used to process data, which allows users to discover information and create new knowledge.

Critical Knowledge and Skills

Knowledge

Students will know:

- DAT-2.A.1 Information is the collection of facts and patterns extracted from data.
- DAT-2.A.2 Data provide opportunities for identifying trends, making connections, and addressing problems.
- DAT-2.A.3 Digitally processed data may show correlation between variables. A correlation found in data does not necessarily indicate that a causal relationship exists. Additional research is needed to understand the exact nature of the relationship.
- DAT-2.A.4 Often, a single source does not contain the data needed to draw a conclusion. It may be necessary to combine data from a variety of sources to formulate a conclusion
- DAT-2.B.1 Metadata are data about data. For example, the piece of data may be an image, while the metadata may include the date of creation or the file size of the image.
- DAT-2.B.2 Changes and deletions made to metadata do not change the primary data.
- DAT-2.B.3 Metadata are used for finding, organizing, and managing information.
- DAT-2.B.4 Metadata can increase the effective use of data or data sets by providing additional information.
- DAT-2.B.5 Metadata allow data to be structured and organized.
- DAT-2.C.1 The ability to process data depends on the capabilities of the users and their tools.
- DAT-2.C.2 Data sets pose challenges regardless of size, such as: § the need to clean data § incomplete data § invalid data § the need to combine data sources
- DAT-2.C.3 Depending on how data were collected, they may not be uniform. For example, if users enter data into an open field, the way they choose to abbreviate, spell, or capitalize something may vary from user to user.
- DAT-2.C.4 Cleaning data is a process that makes the data uniform without changing their meaning (e.g., replacing all equivalent abbreviations, spellings, and capitalizations with the same word).
- DAT-2.C.5 Problems of bias are often created by the type or source of data being collected. Bias is not eliminated by simply collecting more data.
- DAT-2.C.6 The size of a data set affects the amount of information that can be extracted from it.
- DAT-2.C.7 Large data sets are difficult to process using a single computer and may require parallel systems.
- DAT-2.C.8 Scalability of systems is an important consideration when working with data sets, as the

computational capacity of a system affects how data sets can be processed and stored.

- DAT-2.D.1 Programs can be used to process data to acquire information.
- DAT-2.D.2 Tables, diagrams, text, and other visual tools can be used to communicate insight and knowledge gained from data.
- DAT-2.D.3 Search tools are useful for efficiently finding information.
- DAT-2.D.4 Data filtering systems are important tools for finding information and recognizing patterns in data.
- DAT-2.D.5 Programs such as spreadsheets help efficiently organize and find trends in information.
- DAT-2.D.6 Some processes that can be used to extract or modify information from data include the following: § transforming every element of a data set, such as doubling every element in a list, or adding a parent's email to every student record § filtering a data set, such as keeping only the positive numbers from a list, or keeping only students who signed up for band from a record of all the students § combining or comparing data in some way, such as adding up a list of numbers, or finding the student who has the highest GPA § visualizing a data set through a chart, graph, or other visual representation
- DAT-2.E.1 Programs are used in an iterative and interactive way when processing information to allow users to gain insight and knowledge about data
- DAT-2.E.2 Programmers can use programs to filter and clean digital data, thereby gaining insight and knowledge.
- DAT-2.E.3 Combining data sources, clustering data, and classifying data are parts of the process of using programs to gain insight and knowledge from data.
- DAT-2.E.4 Insight and knowledge can be obtained from translating and transforming digitally represented information.
- DAT-2.E.5 Patterns can emerge when data are transformed using programs.
- IOC-1.D.1 Computing innovations can reflect existing human biases because of biases written into the algorithms or biases in the data used by the innovation.
- IOC-1.D.2 Programmers should take action to reduce bias in algorithms used for computing innovations as a way of combating existing human biases.
- IOC-1.D.3 Biases can be embedded at all levels of software development.
- IOC-1.E.1 Widespread access to information and public data facilitates the identification of problems, development of solutions, and dissemination of results.
- IOC-1.E.2 Science has been affected by using distributed and "citizen science" to solve scientific problems.
- IOC-1.E.3 Citizen science is scientific research conducted in whole or part by distributed individuals, many of whom may not be scientists, who contribute relevant data to research using their own computing devices
- IOC-1.E.4 Crowdsourcing is the practice of obtaining input or information from a large number of people via the Internet.
- IOC-1.E.5 Human capabilities can be enhanced by collaboration via computing.
- IOC-1.E.6 Crowdsourcing offers new models for collaboration, such as connecting businesses or social causes with funding.

Skills

Students will be able to:

- DAT-2.A Describe what information can be extracted from data.
- DAT-2.B Describe what information can be extracted from metadata
- DAT-2.C Identify the challenges associated with processing data.
- DAT-2.D Extract information from data using a program.
- DAT-2.E Explain how programs can be used to gain insight and knowledge from data.
- IOC-1.D Explain how bias exists in computing innovations.
- IOC-1.E Explain how people participate in problem-solving processes at scale.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- Code Studio Review and Reflection Questions
- Code.org Activity Guides
- Code Studio Assignments

School Summative Assessment Pan

- Project: Tell a Data Story
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://https://studio.code.org/s/csp9-2022/)

Supplementary Resources

- Blown to Bits (http://www.bitsbook.com/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**

    - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Google Products**
    - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
    - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
    - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❏ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❏ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❏ All assignments have been created in the student's native language.

❏ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❏ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❏ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❏ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.
- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.
- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.
- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.
- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.
- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.
- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.
- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.
- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lesson 1: Learning from Data
- Lesson 2: Exploring One Column
- Lesson 3: Filtering and Cleaning Data
- Lesson 4: Exploring Two Columns
- Lesson 5: Big, Open, and Crowdsourced Data

WEEK 2:

- Lesson 6: Machine Learning
- Lesson 7: Algorithmic Bias
- Lessons 8-9: Project Tell a Data Story Parts 1-2
- Lesson 10: Unit Assessment

Unit 10: Cybersecurity and Global Impacts (2022)

Content Area:	**Applied Tech**
Course(s):	**Generic Course**
Time Period:	**Marking Period 4**
Length:	**3 weeks**
Status:	**Published**

Standards

AP CSP Big Ideas

- BIG IDEA 5: IMPACT OF COMPUTING (IOC) Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand the potential impacts of our programs and be responsible for the consequences. As computer users, we need to understand any potential beneficial or harmful effects and how to protect ourselves and our privacy when using a computer.

Life Literacies and Key Skills

TECH.9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
TECH.9.4.12.CI.2	Identify career pathways that highlight personal talents, skills, and abilities (e.g., 1.4.12prof.CR2b, 2.2.12.LF.8).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).
TECH.9.4.12.DC.1	Explain the beneficial and harmful effects that intellectual property laws can have on the creation and sharing of content (e.g., 6.1.12.CivicsPR.16.a).
TECH.9.4.12.DC.3	Evaluate the social and economic implications of privacy in the context of safety, law, or ethics (e.g., 6.3.12.HistoryCA.1).
TECH.9.4.12.DC.4	Explain the privacy concerns related to the collection of data (e.g., cookies) and generation of data through automated processes that may not be evident to users (e.g., 8.1.12.NI.3).
TECH.9.4.12.DC.5	Debate laws and regulations that impact the development and use of software.
TECH.9.4.12.DC.6	Select information to post online that positively impacts personal image and future college and career opportunities.
TECH.9.4.12.TL.4	Collaborate in online learning communities or social networks or virtual worlds to analyze and propose a resolution to a real-world problem (e.g., 7.1.AL.IPERS.6).
TECH.9.4.12.IML.4	Assess and critique the appropriateness and impact of existing data visualizations for an intended audience (e.g., S-ID.B.6b, HS-LS2-4).

NJ Computer Science Standards

CS.9-12.8.1.12.DA.2	Describe the trade-offs in how and where data is organized and stored.
CS.9-12.8.1.12.DA.5	Create data visualizations from large data sets to summarize, communicate, and support

	different interpretations of real-world phenomena.
CS.9-12.8.1.12.NI.2	Evaluate security measures to address various common security threats.
CS.9-12.8.1.12.NI.3	Explain how the needs of users and the sensitivity of data determine the level of security implemented.
CS.9-12.8.1.12.NI.4	Explain how decisions on methods to protect data are influenced by whether the data is at rest, in transit, or in use.
CS.9-12.8.2.12.EC.1	Analyze controversial technological issues and determine the degree to which individuals, businesses, and governments have an ethical role in decisions that are made.
CS.9-12.8.2.12.EC.3	Synthesize data, analyze trends, and draw conclusions regarding the effect of a technology on the individual, culture, society, and environment and share this information with the appropriate audience.
CS.9-12.8.2.12.ETW.2	Synthesize and analyze data collected to monitor the effects of a technological product or system on the environment.

Transfer Goals

Transfer Goals

Students will be able to independently use their learning to construct a well-rounded and balanced view about cybersecurity in the world around them and to understand global impacts.

Concepts

Essential Questions

- What data are generated by smart phones, and what are they being used for?
- Are innovators responsible for the harmful effects of their computing innovations, even if those effects were unintentional? Why or why not?
- What app or computer software do you use most often and would have a hard time going without? How does this software solve a problem for you or benefit you?

Understandings

- IOC-1 While computing innovations are typically designed to achieve a specific purpose, they may have unintended consequences.
- IOC-2 The use of computing innovations may involve risks to personal safety and identity.

Critical Knowledge and Skills

Knowledge

Students will know:

- IOC-1.A.1 People create computing innovations.
- IOC-1.A.2 The way people complete tasks often changes to incorporate new computing innovations.
- IOC-1.A.3 Not every effect of a computing innovation is anticipated in advance.
- IOC-1.A.4 A single effect can be viewed as both beneficial and harmful by different people, or even by the same person.
- IOC-1.A.5 Advances in computing have generated and increased creativity in other fields, such as medicine, engineering, communications, and the arts.
- IOC-1.B.1 Computing innovations can be used in ways that their creators had not originally intended: § The World Wide Web was originally intended only for rapid and easy exchange of information within the scientific community. § Targeted advertising is used to help businesses, but it can be misused at both individual and aggregate levels. § Machine learning and data mining have enabled innovation in medicine, business, and science, but information discovered in this way has also been used to discriminate against groups of individuals.
- IOC-1.B.2 Some of the ways computing innovations can be used may have a harmful impact on society, the economy, or culture.
- IOC-1.B.3 Responsible programmers try to consider the unintended ways their computing innovations can be used and the potential beneficial and harmful effects of these new uses.
- IOC-1.B.4 It is not possible for a programmer to consider all the ways a computing innovation can be used.
- IOC-1.B.5 Computing innovations have often had unintended beneficial effects by leading to advances in other fields.
- IOC-1.B.6 Rapid sharing of a program or running a program with a large number of users can result in significant impacts beyond the intended purpose or control of the programmer.
- IOC-2.A.1 Personally identifiable information (PII) is information about an individual that identifies, links, relates, or describes them. Examples of PII include: § Social Security number § age § race § phone number(s) § medical information § financial information § biometric data
- IOC-2.A.10 Commercial and governmental curation of information may be exploited if privacy and other protections are ignored.
- IOC-2.A.11 Information placed online can be used in ways that were not intended and that may have a harmful impact. For example, an email message may be forwarded, tweets can be retweeted, and social media posts can be viewed by potential employers.
- IOC-2.A.12 PII can be used to stalk or steal the identity of a person or to aid in the planning of other criminal acts.
- IOC-2.A.13 Once information is placed online, it is difficult to delete.
- IOC-2.A.14 Programs can collect your location and record where you have been, how you got there, and how long you were at a given location.

- IOC-2.A.15 Information posted to social media services can be used by others. Combining information posted on social media and other sources can be used to deduce private information about you.
- IOC-2.A.2 Search engines can record and maintain a history of searches made by users.
- IOC-2.A.3 Websites can record and maintain a history of individuals who have viewed their pages.
- IOC-2.A.4 Devices, websites, and networks can collect information about a user's location.
- IOC-2.A.5 Technology enables the collection, use, and exploitation of information about, by, and for individuals, groups, and institutions.
- IOC-2.A.6 Search engines can use search history to suggest websites or for targeted marketing.
- IOC-2.A.7 Disparate personal data, such as geolocation, cookies, and browsing history, can be aggregated to create knowledge about an individual.
- IOC-2.A.8 PII and other information placed online can be used to enhance a user's online experiences.
- IOC-2.A.9 PII stored online can be used to simplify making online purchases.
- IOC-2.B.1 Authentication measures protect devices and information from unauthorized access. Examples of authentication measures include strong passwords and multifactor authentication.
- IOC-2.B.10 All real-world systems have errors or design flaws that can be exploited to compromise them. Regular software updates help fix errors that could compromise a computing system.
- IOC-2.B.11 Users can control the permissions programs have for collecting user information. Users should review the permission settings of programs to protect their privacy.
- IOC-2.B.2 A strong password is something that is easy for a user to remember but would be difficult for someone else to guess based on knowledge of that user.
- IOC-2.B.3 Multifactor authentication is a method of computer access control in which a user is only granted access after successfully presenting several separate pieces of evidence to an authentication mechanism, typically in at least two of the following categories: knowledge (something they know), possession (something they have), and inherence (something they are).
- IOC-2.B.4 Multifactor authentication requires at least two steps to unlock protected information; each step adds a new layer of security that must be broken to gain unauthorized access.
- IOC-2.B.5 Encryption is the process of encoding data to prevent unauthorized access. Decryption is the process of decoding the data. Two common encryption approaches are: § Symmetric key encryption involves one key for both encryption and decryption. § Public key encryption pairs a public key for encryption and a private key for decryption. The sender does not need the receiver's private key to encrypt a message, but the receiver's private key is required to decrypt the message.
- IOC-2.B.6 Certificate authorities issue digital certificates that validate the ownership of encryption keys used in secure communications and are based on a trust model.
- IOC-2.B.7 Computer virus and malware scanning software can help protect a computing system against infection.
- IOC-2.B.8 A computer virus is a malicious program that can copy itself and gain access to a computer in an unauthorized way. Computer viruses often attach themselves to legitimate programs and start running independently on a computer.
- IOC-2.B.9 Malware is software intended to damage a computing system or to take partial control over its operation.
- IOC-2.C.1 Phishing is a technique that attempts to trick a user into providing personal information. That personal information can then be used to access sensitive online resources, such as bank accounts and emails.
- IOC-2.C.2 Keylogging is the use of a program to record every keystroke made by a computer user in order to gain fraudulent access to passwords and other confidential information.
- IOC-2.C.3 Data sent over public networks can be intercepted, analyzed, and modified. One way that this can happen is through a rogue access point.

- IOC-2.C.4 A rogue access point is a wireless access point that gives unauthorized access to secure networks.
- IOC-2.C.5 A malicious link can be disguised on a web page or in an email message.
- IOC-2.C.6 Unsolicited emails, attachments, links, and forms in emails can be used to compromise the security of a computing system. These can come from unknown senders or from known senders whose security has been compromised.
- IOC-2.C.7 Untrustworthy (often free) downloads from freeware or shareware sites can contain malware.

Skills

Students will be able to:

- IOC-1.A Explain how an effect of a computing innovation can be both beneficial and harmful.
- IOC-1.B Explain how a computing innovation can have an impact beyond its intended purpose
- IOC-2.A Describe the risks to privacy from collecting and storing personal data on a computer system.
- IOC-2.B Explain how computing resources can be protected and can be misused.
- IOC-2.C Explain how unauthorized access to computing resources is gained.

Assessment and Resources

School Formative Assessment Plan (Other Evidence)

- "Unplugged" Classroom Tasks
- Code Studio Review and Reflection Questions
- Code.org Activity Guides
- Code Studio Assignments

School Summative Assessment Pan

- Project Innovation Simulation
- Unit Assessment (using LinkIt)

Primary Resources

- Code.org AP CSP Curriculum (https://https://studio.code.org/s/csp10-2022/)

Supplementary Resources

- Blown to Bits (http://www.bitsbook.com/)

Technology Integration and Differentiated Instruction

Technology Integration

- **Code.org Code Studio**
  - Code Studio provides videos, readings, and review and reflection questions that students can go through at their own pace.
- **Google Products**
  - Google Classroom - Used for daily interactions with the students covering a vast majority of different educational resources (Daily Notes, Exit Tickets, Classroom Polls, Quick Checks, Additional Resources/ Support, Homework, etc.)
  - GAFE (Google Apps For Education) - Using various programs connected with Google to collaborate within the district, co-teachers, grade level partner teacher, and with students to stay connected with the content that is covered within the topic. Used to collect data in real time and see results upon completion of the assignments to allow for 21st century learning.
- **One to One laptops**
  - All students within the West Deptford School District are given a computer, allowing for 21st century learning to occur within every lesson/topic.

Differentiated Instruction

Gifted Students (N.J.A.C.6A:8-3.1)

❏ Within each lesson, the Gifted Students are given choice on topic and subject matter allowing them to explore interests appropriate to their abilities, areas of interest and other courses.

English Language Learners (N.J.A.C.6A:15)

❏ Within each lesson, the English Language Learners are given choice of topic and resources so that their materials are within their ability to grasp the language.

❏ All assignments have been created in the student’s native language.

❏ Work with ELL Teacher to allow for all assignments to be completed with extra time.

At-Risk Students (N.J.A.C.6A:8-4.3c)

❑ Within each lesson, the at-risk students are given choice of topic and resources so that their materials are within their ability level and high-interest.

Special Education Students (N.J.A.C.6A:8-3.1)

❑ Within each lesson, special education students are given choice of topic and resources so that their materials are within their ability level and high-interest.

❑ All content will be modeled with examples and all essays are built on a step-by-step basis so modifications for assignments in small chunks are met.

All other IEP modifications will be honored (ie. hard copies of notes, directions restated, etc.)

Interdisciplinary Connections

- Mathematics: Computer science heavily relies on mathematical concepts, such as logic, algorithms, and data structures. You can explore mathematical foundations of computer science, including binary numbers, Boolean algebra, and graph theory.

- Physics: Physics and computer science share common principles, such as modeling and simulation. You can examine how computers are used to simulate physical systems, study computational physics, or explore the role of algorithms in solving physics problems.

- Biology: Bioinformatics is an interdisciplinary field that combines computer science and biology. You can explore how computational methods are used to analyze biological data, study genetics, or simulate biological processes.

- Economics: The field of computational economics uses computer models and algorithms to study economic systems. You can explore topics such as market simulations, algorithmic trading, or the impact of technology on the economy.

- Environmental Science: Computer science can be applied to environmental science for tasks like data analysis, modeling climate patterns, or studying the impact of human activities on the environment. You can explore how computer science tools are used in environmental research and conservation efforts.

- Social Sciences: Computer science intersects with various social sciences, including sociology, psychology, and political science. You can explore the social implications of technology, ethics in computing, or analyze large-scale social networks using computational methods.

- Art and Design: Computer science and art can be connected through areas like digital art, computer graphics, and interactive installations. You can explore creative coding, visual design principles, or study the intersection of technology and artistic expression.

- Language and Linguistics: Natural language processing, a subfield of computer science, focuses on the interaction between computers and human language. You can explore topics like text analysis, speech recognition, or machine translation.

- History: You can study the historical development of computing technologies, the evolution of programming languages, or the impact of computing on historical events and society.
- Engineering and Technology: Computer science is closely tied to engineering and technology. You can explore hardware and software engineering concepts, robotics, or delve into the principles of computer architecture.

Learning Plan / Pacing Guide

WEEK 1:

- Lesson 3: Data Policies and Privacy
- Lesson 4: The Value of Privacy
- Lesson 6: Security Risks Part 1
- Lesson 7: Security Risks Part 2
- Lesson 9: Protecting Data Part 1

WEEK 2:

- Lesson 10: Protecting Data Part 2
- Lesson 14: Unit Assessment

****BREAK FOR AP EXAM REVIEW****

POST EXAM

- Project Innovation Simulation