2024
Computer Science Standards of Learning

Grade 6 Instructional Guide

VIRGINIA DEPARTMENT OF EDUCATION

Virginia Department of Education
P.O. Box 2120
Richmond, Virginia 23218-2120
http://www.doe.virginia.gov

Superintendent of Public Instruction
Emily Anne Gullickson

Assistant Superintendent of Teaching and Learning
Michelle Wallace, Ph.D.

Office of Educational Technology and Classroom Innovation
Calypso Gilstrap, Associate Director
Keisha Tennessee, Computer Science Coordinator

Table of Contents

2024 COMPUTER SCIENCE *STANDARDS OF LEARNING*

Introduction

Virginia's Computer Science standards aim to raise our aspirations for computational instruction to enable students to engage and thrive in a digital world. Beginning in the earliest grades and continuing through 12th grade, students must develop a foundation of computer science knowledge and learn new approaches to problem solving that harness the power of computational thinking to become both users and creators of computing technology.

It is important for every student to engage in computer science education from the earliest ages. This early and sustained access equips students with foundational problem-solving practices, develops their understanding of how current and emerging computer science technologies work, and fosters curiosity, interest, and innovation with computer science.

Foundational Principles

Computer Literacy is foundational to learning and post-secondary success as technology becomes increasingly incorporated into all aspects of everyday life. Computer Literacy provides critical knowledge and skills for all subject areas including mathematics, science, history, English, and fine arts. By applying computer science as a tool for learning and expression in a variety of disciplines and interests, students will actively and proficiently participate in a world that is increasingly influenced by digital technology.

Computer Science fosters problem solving skills that are essential to all educational disciplines and post-secondary employment opportunities. Understanding how multi-step solutions are executed within computer programs allows students the opportunity to use metacognitive strategies with tasks they are performing as they work and study in any topic area. Computer Science should become an essential part of Virginia K-12 education, accessible by all, rather than a vocational part of education only for those headed to technology-based employment.

Computer Science instruction must maintain the pace of technology evolution to prepare students for the workforce. Computer science is a core technology component for students to have the ability to adapt to the future evolution of work. The workforce of the future will increasingly require that all adults effectively work in digital environments and utilize technology both ethically and responsibly. As a result, we must prioritize preparing all students with integral computer science learning opportunities throughout their academic career to ensure they are prepared for a post-secondary success in a digital world that includes computer-based problem solving, artificial intelligence and communication rooted in the use of digital tools.

Students should gain specific digital and computational concepts to harness the power of computer science and derivative applications, such as machine learning, online programming, virtual reality, and Artificial Intelligence (AI), to embrace innovation and chart the future of individuals, business, and government responsibly.

Instructional Intent and Integration

Computer science is an academic discipline that encompasses both conceptual foundations and applied practices. It can be taught effectively with or without computing devices, as many key skills, such as logical reasoning, pattern recognition, decomposition, and sequencing can be developed through with or without a computing device.

In primary grades, overlapping concepts between computer science and other content areas may be taught within the same instructional context. When doing so, it is essential that educators intentionally align instruction to ensure that the full intent and specifications of the computer science standard are addressed, even when the learning experience is shared with another content area.

As students' progress into upper elementary and beyond, instruction should be explicit, ensuring students are able to identify and understand the computer science concepts and practices embedded within those shared experiences. By naming the connections and calling out the domain specific elements of computer science, students can deepen their disciplinary understanding, build metacognitive awareness, and transfer their knowledge and skills across contexts.

It is important to recognize that not all computer science concepts will naturally overlap with other subjects. Concepts such as algorithms, data representation, networks, and programming require dedicated instructional time and may be taught independently of other content areas. Whether through integration or stand-alone instruction, computer science should be approached with the same level of intentionality and rigor as other academic subjects, ensuring students develop a coherent and comprehensive understanding from kindergarten through grade 12.

Sixth Grade: 2024 Computer Science *Standards of Learning*

In the Sixth Grade, students build upon their understanding of computing systems and the network and gather and share information effectively. Students gain a more defined understanding of how data is transmitted over the Internet. Students explore real-world applications of cybersecurity through exploring laws governing privacy and understanding user agreements. Students continue to use computational thinking and an iterative design process to create algorithms and programs. A greater emphasis on the application of data and analysis is present with an understanding of the importance of accuracy and reliability of data, and the use of appropriate computational tools to create data visualizations that are used to make predictions and draw conclusions. Students actively engage in troubleshooting and documenting hardware and software issues to foster resilience, analytical reasoning, and effective communication. Students explore potential career paths and evaluate how computer science skills can be applied in various professions.

Algorithms and Programming (AP)

6.AP.1 The student will apply computational thinking to identify patterns, make use of decomposition to break down problems or processes into sub-components, and design algorithms.

a. Identify patterns and repeated steps in an algorithm, problem, or process.
b. Decompose an algorithm, problem, or process into sub-components.
c. Abstract relevant information to identify essential details.
d. Design algorithms using abstraction to accomplish a task or express a computational process.

6.AP.2 The student will plan and implement algorithms that include conditional control structures and collection of numeric data using a block-based or text-based tool.

a. Create a decision tree diagram to illustrate the decision-making process.
b. Read and write programs that initialize Boolean, integer, and decimal number variables.
c. Read and write programs that collect numeric data from users.
d. Read and write programs that contain nested conditional control structures.
e. Predict the results of logic expressions that use Boolean operators: and, or, and not; including expressions that use relational expressions as one or more operands.

6.AP.3 The student will use the iterative design process to create, test, and debug programs using a block-based or text-based programming language.

a. Create and test programs that use multiple conditional control structures.
b. Incorporate existing code, media, or libraries into original programs.
c. Trace and predict outcomes of programs.
d. Analyze and describe program results to assess validity of outcomes.
e. Analyze the outcomes of programs to identify logic and syntax errors.
f. Incorporate feedback from others to refine program.
g. Revise and improve programs to resolve errors and produce desired outcomes.

6.AP.4 The student will demonstrate proper attribution when incorporating ideas and works of others.

a. Identify and give proper attribution of information and assets from the Internet and other sources.

Computing Systems (CSY)

6.CSY.1 The student will define and explain application software and operating systems of a computing device within a computing system.

a. Define and describe the functions of an operating system and application software.
b. List advantages and limitations of application software and operating systems based on the needs of the user.

6.CSY.2 The student will identify and explain hardware, software, and connectivity problems and troubleshooting solutions.

a. Identify and explain hardware, software, and connectivity problems and solutions with accurate terminology.
b. Identify resources for troubleshooting hardware, software, and connectivity-related problems.

6.CSY.3 The student will identify and describe Artificial Intelligence (AI).

a. Define artificial intelligence and identify the characteristics of artificial intelligence.
b. Describe how AI technologies generate information or automate decision and how people interact with AI technologies.
c. Define algorithmic bias and explain its consequences on AI technologies and systems.

Cybersecurity (CYB)

6.CYB.1 The student will evaluate the risks and benefits associated with sharing personal and public resources or artifacts.

a. Identify and explain the difference between personal and public information.
b. Discuss the consequences of sharing personal and confidential information online.
c. Evaluate risks and benefits associated with sharing information online.

6.CYB.2 The student will investigate various usage agreements designed to protect individuals.

a. Identify laws governing privacy with computing devices and emerging technologies.
b. Investigate and describe common components of usage agreements.
c. Identify user and company protections in a usage agreement.

Data and Analysis (DA)

6.DA.1 The student will utilize computational tools to collect and organize data.

a. Select and use appropriate computational tools to collect data.
b. Organize data to make it easier to understand and use.
c. Clean data to remove and correct errors.
d. Analyze data sources for accuracy and reliability.

6.DA.2 The student will utilize computational tools to visualize and evaluate data.

a. Identify different types of visual representations of data.
b. Compare various visual representations and identify when each should be used.
c. Create charts, graphs, models, and simulations to visualize data.
d. Describe and synthesize information from a visual representation of data.

6.DA.3 The student will make predictions and draw conclusions from data visualizations.

a. Visualize data using appropriate graphs, charts, and data visualization techniques to enhance understanding and communicate findings effectively.
b. Use computational tools to analyze patterns within data sets and identify trends.
c. Draw conclusions and make predictions based on the analysis and interpretation of the data visualization.
d. Utilize simulations and models to formulate, refine, and test hypotheses.

6.DA.4 The student will identify ways people curate and provide training data.

a. Identify and list ways people provide data that is used as training data.
b. Discuss the role of human intervention in curating training data.
c. Identify and describe the effect training data has on the accuracy of artificial intelligence systems.

Impacts of Computing (IC)

6.IC.1 The student will assess the impact of computing technologies on local society.

a. Explain how computing impacts innovation and describe the development of new computing technologies in communication, entertainment, and business.
b. Discuss how computing technologies have influenced various industries and sectors locally.
c. Research simple and complex problems that computing systems can be used to solve.
d. Analyze the implications of emerging technologies and potential real-world impact in the local community.

6.IC.2 The student will analyze the impact of screen time on physical and mental health.

a. Analyze and describe the impact of excessive technology usage may have on one's physical health.
b. Examine the impact of blue light on sleep patterns and regulations.
c. Propose strategies that provide alternatives of technology usage to promote physical activity.
d. Discuss the potential impact the use of social media may have on self-identity and mental health.
e. Define cyberbullying and its impact on one's health and well-being.
f. Discuss the possible effects of cyberbullying.
g. Identify ways to report illegal or psychologically maladaptive online behavior.

6.IC.3 The student will explore career pathways and identify how computer science and computational thinking practices align with these pathways.

a. Investigate a career of interest and determine how computer science and computational thinking practices are used in the chosen career.

6.IC.4 The student will identify copyrighted and licensed software material.

a. Identify the role of software licenses, including open-source, and why they are used.
b. Compare and contrast the positives and negatives of various software licenses.

6.IC.5 The student will describe the impacts of computing network architecture, including the role of the Internet in society.

a. Discuss ethical issues and laws related to accessibility, censorship, privacy, access, and safety while using the Internet.
b. Explain the role broadband connectivity has in social life, culture, and global economy.

6.IC.6 The student will investigate and analyze the impact of the progression and advancement of AI technologies on industries.

a. Discuss the type of industries that may be impacted by the use and integration of Artificial Intelligence (AI).
b. Compare and contrast the evolving nature of work across diverse industries because of the progression and advancement of Artificial Intelligence.

Networks and the Internet (NI)

6.NI.1 The student will outline the advantages and disadvantages of transmitting information over the Internet, including speed, reliability, cost, and security.

a. Explain the role of the Internet in social life, culture, and the economy.
b. Explain data transfer and the impact of connectivity speed when data is going from one device to another.
c. Compare the speed and reliability of various data transmission media.
d. Describe the advantages and disadvantages of transporting information over the Internet.

Computer Science

Instructional Guide

This instructional guide, a companion document to the 2024 Computer Science *Standards of Learning*, amplifies each standard by defining the core knowledge and skills in practice, supporting teachers and their instruction, and serving to transition classroom instruction from the 2017 Computer Science *Standards of Learning* to the newly adopted standards.

Computer Science Instructional Guide Framework

This instructional guide includes, Understanding the Standard, Concepts and Connections, Opportunities for Integration, and Skills in Practice aligned to each standard. The purpose of each is explained.

Understanding the Standard

This section is designed to unpack the standards, providing both students and teachers with the necessary knowledge to support effective instruction. It includes core concepts that students are expected to learn, as well as background knowledge that teachers can use to deepen their understanding of the standards and plan standards-aligned lessons.

Concepts and Connections

This section outlines concepts that transcend grade levels and thread through the K through 12 computer science as appropriate at each level. Concept connections reflect connections to prior grade-level concepts as content and practices build within the discipline as well as potential connections across disciplines. The connections across disciplines focus on direct standard alignment, where concepts and practices in computer science overlap with similar ideas in other disciplines.

Computer Science connections are aligned to the: 2024 English *Standards of Learning*, 2023 History and Social Science, 2023 Mathematics *Standards of Learning*, 2020 Digital Learning Integration *Standards of Learning*, and 2018 Science *Standards of Learning*.

These cross-disciplinary concepts and practices are foundational for effective interdisciplinary integration.

Opportunities for Integration

This section provides lesson ideas for integrating computer science with English, history and social science, mathematics, and science through multidisciplinary, interdisciplinary, and transdisciplinary approaches. Lesson ideas may involve the integration of standards that may or may not be directly aligned yet are strategically taught together to achieve a purposeful and authentic learning experience that supports meaningful student outcomes such as deeper understanding, skill transfer, and real-world application.

Skills in Practice

This section focuses on instructional strategies that teachers can use to develop students' skills, deepen their conceptual understanding, and encourage critical thinking. These practices are designed to support curriculum writers and educators in weaving pedagogical approaches that deepen student understanding of unit and course objectives, ultimately enhancing learning outcomes. This section provides a framework for planning effective and engaging lessons.

Sixth Grade: Computer Science Instructional Guide

In the Sixth Grade, students build upon their understanding of computing systems and the network and gather and share information effectively. Students gain a more defined understanding of how data is transmitted over the Internet. Students explore real-world applications of cybersecurity through exploring laws governing privacy and understanding user agreements. Students continue to use computational thinking and an iterative design process to create algorithms and programs. A greater emphasis on the application of data and analysis is present with an understanding of the importance of accuracy and reliability of data, and the use of appropriate computational tools to create data visualizations that are used to make predictions and draw conclusions. Students actively engage in troubleshooting and documenting hardware and software issues to foster resilience, analytical reasoning, and effective communication. Students explore potential career paths and evaluate how computer science skills can be applied in various professions.

Algorithms and Programming (AP)

6.AP.1 The student will apply computational thinking to identify patterns, make use of decomposition to break down problems or processes into sub-components, and design algorithms.

a. Identify patterns and repeated steps in an algorithm, problem, or process.
b. Decompose an algorithm, problem, or process into sub-components.
c. Abstract relevant information to identify essential details.
d. Design algorithms using abstraction to accomplish a task or express a computational process.

Understanding the Standard

Computational thinking (CT) is a structured, solution-driven problem-solving approach that leverages logical reasoning and systematic analysis to address complex challenges. It often results in the creation of a computational artifact or the implementation of an algorithmic process. Computational thinking is universally applicable across disciplines that can be used to deconstruct complex problems, identify essential patterns, and formulate efficient, scalable solutions. Within computer science, computational thinking serves as a foundational skill, underpinning algorithm design, data analysis, and the development of technology-driven solutions to real-world problems. Computational thinking consists of decomposition, pattern analysis, abstraction, and algorithmic thinking.

- Decomposition is the problem-solving practice of breaking apart a problem or process into subcomponents. This involves identifying and recognizing relationships among the parts.
- Abstraction is the problem-solving practice of representing and simplifying relevant information to identify essential details. This involves hiding less important details.

- Pattern analysis is the problem-solving practice of recognizing commonalities, differences, and predictable relationships like sequences.

- Algorithmic thinking is the problem-solving practice of sequencing information.

Computational thinking (CT) is universally applicable across various fields, allowing individuals to break down complex problems and develop efficient solutions. Its role in computer science is particularly important, as it serves as the foundation for designing algorithms, analyzing data, and solving real-world challenges through technology.

Computational thinking is evident when students approach problem-solving by employing strategies such as:
- Breaking a problem or task into smaller steps or manageable parts(decomposition),
- Identifying repeated, reoccurring actions or characteristics(patterns),
- Focusing on key information and disregarding irrelevant details (abstraction),
- Creating step-by-step instructions or algorithms to solve a problem or task and the ability to compare different methods for efficiency (algorithm comparison).

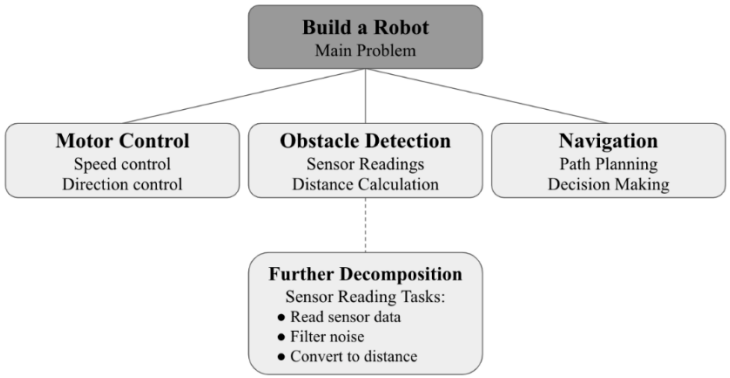
Teachers should intentionally design learning experiences that foster the practical application of computational thinking, such as hands-on programming projects and complex problem-solving tasks that reflect authentic, real-world contexts. The creation of meaningful, developmentally appropriate tasks accompanied by differentiated levels of challenges to ensure that all students can engage with and advance their computational thinking competencies. Assessment practices should prioritize the evaluation of students' cognitive processes, strategies, and metacognitive reflections rather than focusing solely on the final product. Structured collaborative activities not only cultivate teamwork and communication skills but also simulate authentic practices within professional computing environments. Furthermore, integrating computational thinking across disciplinary boundaries promotes deeper conceptual understanding and highlights its interdisciplinary relevance.

[6.AP.1a] Pattern identification involves analyzing an algorithm, problem, or process to detect recurring structures, sequences, or behaviors. A pattern refers to a recurring sequence or set of operations that exhibit predictable repetition. This concept is often exemplified through iterative control structures such as loops, which enable the execution of a code block multiple times based on defined conditional expressions. Recognizing these patterns supports the abstraction and generalization of solutions, allowing for the elimination of redundancy, enhancement of algorithmic efficiency, and development of reusable components. Instruction should emphasize how loop constructs, conditional logic, and control flow structures operationalize pattern recognition in practice, optimizing problem-solving strategies and enabling scalable, dynamic computation.

[6.AP.1b] Decomposition is the systematic process of breaking down a complex algorithm, problem, or process into smaller, more manageable sub-components. This modular approach enhances computational efficiency and clarity by enabling focused analysis and incremental solution

development for each part. Rather than attempting to address the entire problem holistically, each functional unit or procedural element is isolated, examined, and addressed individually. This stepwise strategy improves computational efficiency, supports parallel development, and lays the groundwork for designing scalable, maintainable algorithms.

- Decomposition example: When designing a robot to navigate a maze, the overall task is too complex to solve holistically. To manage this complexity, the problem is decomposed into distinct subcomponents such as detecting walls, controlling motor movement, processing sensor input, and determining navigation paths. Each module represents a focused task within the broader system. This structured breakdown allows algorithmic development for each subcomponent to proceed independently, improving clarity, debugging, and eventual system integration. Through decomposition, the problem is made tractable and aligned with modular design principles essential in computer science.



[6.AP.1c] Abstraction is the cognitive process of distilling a problem or system to its most critical elements by filtering out extraneous or non-essential information. This practice reduces complexity and enhances clarity, allowing for the development of simplified representations or models that retain only the features relevant to the task at hand. By focusing on high-level functionality rather than low-level implementation details, abstraction enables more efficient problem-solving, facilitates the design of generalized algorithms, and supports the creation of modular, reusable components.

- Abstraction example: Encryption in cybersecurity exemplifies abstraction by concealing the underlying complexity of cryptographic algorithms. Through this process, readable information (plaintext) is transformed into an unreadable format (ciphertext) using mathematically sophisticated procedures. End users interact with encryption tools such as secure messaging apps or file protection systems—without needing to comprehend the algorithmic details or cryptographic protocols involved. This abstraction enables the application of advanced security measures while simplifying user interaction, illustrating how abstraction supports both usability and robust digital protection in modern computing systems.

[6.AP.1d] The ability to design algorithms using abstraction requires one to identify the essential information needed to complete the task and omit unnecessary details. This abstraction enables the creation of structured, high-level representations such as pseudocode, flowcharts, or modular functions that illustrate the logic of the process without relying on low-level implementation specifics. Teachers can guide students to define key inputs, outputs, and conditions, then organize that distilled information into a complete and logically ordered sequence of steps. This approach simplifies complex problems and supports the development of scalable, adaptable, and reusable solutions. Algorithm design through abstraction promotes modularity, enhances readability, and aligns with best practices in computational problem-solving and software engineering, such as hierarchical decomposition and separation of concerns.

Concepts and Connections

Concepts

Computational thinking is a structured approach to problem-solving that applies logical reasoning and systematic analysis to address complex challenges. It involves identifying patterns and repeated steps in processes, decomposing problems into smaller, manageable components, and abstracting essential details to reduce complexity, and designing algorithmic solutions using formal representations such as pseudocode, decision trees, and flowcharts to clearly define logic and control flow.

Connections

Within the grade level/course: At this grade level, students engage in computational thinking practices such as decomposition, pattern recognition, abstraction, and algorithmic thinking to design programs for creative expression or scientific exploration. (6.AP.1).
Vertical Progression: In Grade 5, students apply abstraction as they design an algorithm to solve a problem (5.AP.1). In Grade 7, students expand on abstraction by designing algorithms for the purposes of accomplishing a task for creative expression or scientific exploration. (7.AP.1).

Across Content Areas

Mathematics

- **6.CE.1** The student will estimate, demonstrate, solve, and justify solutions to problems using operations with fractions and mixed numbers, including those in context.
- **6.CE.2** The student will estimate, demonstrate, solve, and justify solutions to problems using operations with integers, including those in context.
- **6.NS.3** The student will recognize and represent patterns with whole number exponents and perfect squares.
- **6.PFA.1** The student will use ratios to represent relationships between quantities, including those in context.

Science

- **6.1** The student will demonstrate an understanding of scientific engineering practices by a) asking questions to determine relationships between independent and dependent variables, b) planning and carrying out investigations, c) interpreting, analyzing and evaluating data, d) constructing and critiquing conclusions and explanations, e) developing and using models, f) obtaining, evaluating and communicating information. *6.1 standard is integrated within science content and not taught in isolation.*
- **6.2** The student will investigate and understand that the solar system is organized and the various bodies in the solar system interact.
- **6.3** The student will investigate and understand that there is a relationship between the sun, Earth, and the moon.
- **6.4** The student will investigate and understand that there are basic sources of energy and that energy can be transformed.

Digital Learning Integration

- **6-8.EL** Students leverage technologies, including assistive technologies, to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.
  C. Students seek feedback from people, including peers, teachers, staff familiar with assistive technologies, and functionalities embedded in technologies to make changes to improve and demonstrate their learning.

Opportunities for Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students analyze a literary text by identifying key plot elements: exposition, rising action, climax, falling action, and resolution and represent these through an algorithm or flowchart that illustrates the logical progression of events.

Mathematics

- Students apply abstraction to analyze arithmetic and geometric sequences by identifying underlying patterns and relationships. They then design algorithms or equations to calculate missing terms and predict future values, strengthening both their problem-solving and computational reasoning skills.
- Students explore how everyday tasks (like brushing teeth or posting a photo online) involve mathematical thinking. Through structured analysis of algorithms, students can model multistep processes, optimize decision-making structures, and evaluate efficiency of step-based problem solving.

Science

- students explore different sources of water pollution in their local community or region. Through case studies, image analysis, and data interpretation, they'll practice the skill of abstraction: identifying essential contributors, patterns of contamination, and credible data sources

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

1. Decompose Real-World Problems
2. Explore Common Features and Relationships to Extract Patterns
3. Apply Abstraction to Simplify, Represent, and Problem Solve
4. Apply Algorithmic Thinking to Problem Solve and Create
5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

C. Fostering Iterative Design Practices:

1. Decompose Real-World Problems
2. Plan and Design Artifacts
3. Create, Communicate, and Document Solutions
4. Test and Optimize Artifacts

6.AP.2 The student will plan and implement algorithms that include conditional control structures and collection of numeric data using a block-based or text-based tool.

a. Create a decision tree diagram to illustrate the decision-making process.
b. Read and write programs that initialize Boolean, integer, and decimal number variables.
c. Read and write programs that collect numeric data from users.
d. Read and write programs that contain nested conditional control structures.
e. Predict the results of logic expressions that use Boolean operators: and, or, and not; including expressions that use relational expressions as one or more operands.

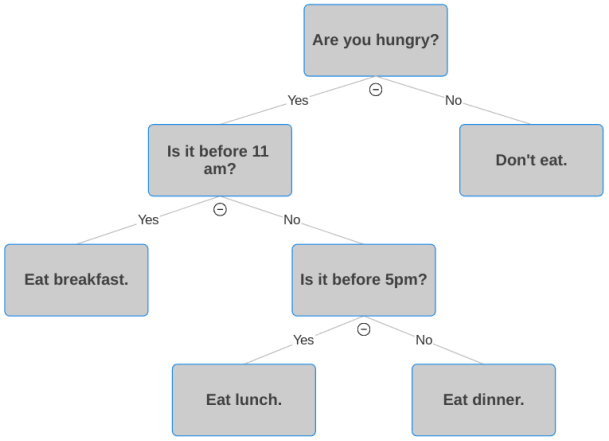
Understanding the Standard

At the elementary level, students begin developing foundational programming skills through both the design and implementation of algorithms. Working collaboratively and independently, they create programs that model familiar, real-world tasks. As their proficiency grows, they begin incorporating fundamental control structures such as loops and events, allowing for more sophisticated and efficient program behavior. In middle school, students extend these skills by using block-based or text-based programming environments, enabling them to design, test, and refine increasingly complex algorithms and computational solutions.In middle school students continue to build and expand these skills using block-based or text-based tools.

[6.AP.2a] Programs are collections of code organized in algorithms that can accomplish a variety of tasks. Programs can be developed to perform calculations, manipulate data, or to be creative. Control structures, such as conditionals and loops, are integrated into programs to allow for interactivity, decision-making, and user input.

A decision tree is a type of flowchart used to represent the logic of decision-making processes in a clear, visual format. It begins with a starting condition or question, represented by a node, and branches into multiple paths based on possible outcomes or choices. Each internal node corresponds to a decision or test (e.g., Is the number greater than 10?), while each branch represents a possible answer (e.g., Yes or No), and each leaf node indicates a resulting action or output.

Decision trees illustrate how a program evaluates different conditions and responds accordingly, making abstract control flow concepts, such as if-then-else statements more concrete for students. Teachers can guide students in constructing decision trees by identifying key conditions in a problem, mapping out all possible outcomes, and visually organizing them to reflect the program's logic. This process supports algorithmic thinking by emphasizing the structured reasoning behind program behavior.
Example below:



[6.AP.2b] Data types define the form and characteristics of data stored and processed in a program. Boolean data types represent true or false values used in logical expressions. Integer data types store whole numbers without fractional parts. Decimal, or floating-point, data types represent numbers with fractional components, supporting a wide range of real-number values.

- Boolean data types represent binary states and hold only two possible values: true or false. In most programming languages, these are often implemented using binary representations, where 1 denotes true and 0 denotes false. Boolean variables are commonly used in conditional statements and logic operations, such as determining whether a condition is met or controlling the flow of a program (e.g., if statements, loops).
- Integer data types represent whole numbers, both positive and negative, without any fractional component. They are typically used in programs where precise counting, indexing, or discrete numeric values are required, for example, tracking the number of iterations in a loop or storing a user's age.
- Decimal data types, also referred to as floating-point numbers, are used to represent numbers that include a fractional component (e.g., 3.14, -0.75). These are essential in situations that require greater precision, such as scientific calculations, financial data processing, or measurement-based applications.

The choice of data type depends on the nature of the data collected and the intended operations within the program. Boolean variables are ideal for representing binary conditions (e.g., on/off, win/lose, yes/no), while integers and decimals are used when working with numeric values, particularly when arithmetic operations or multi-state data modeling are involved. Selecting the appropriate data type is essential for efficient memory usage, accurate computation, and correct program behavior.

[6.AP.2c] Numeric data types represent values that can be processed through arithmetic and logical operations. These include integers, which are whole numbers without decimal components, and floating-point numbers, which include fractional parts and support representation of real numbers.

Programming languages use numeric data types to enable operations such as addition, subtraction, multiplication, division, and comparison. These types are foundational in implementing algorithms, controlling program logic, and managing data that requires quantitative analysis.

[6.AP.2d] Control structures such as conditionals and loops determine the order in which instructions are executed in a program. Nested conditionals and nested loops extend these control structures by placing one conditional or loop inside another, allowing for more complex, layered decision-making and repetition. Nested conditionals evaluate multiple criteria in a sequence, where the outcome of one decision influences the next. Nested loops perform repeated operations within other loops, enabling iteration over multidimensional data or executing a series of operations multiple times within a defined structure

Nested conditionals are an "if-else" statement inside another "if-else" statement. They allow more complex decision-making processes. They are used to evaluate multiple criteria within another conditional statement, enabling more complex decision-making.
Example: if temperature > 80: if weather == "Sunny": print("Go to the beach.") else: print("Stay indoors.") else: print("Wear a jacket.")

Nested conditionals are control structures in programming that evaluate one conditional statement within another. They support decision-making processes that depend on the outcome of a prior condition. Each nested level introduces a new layer of logical evaluation that determines the program's flow based on multiple interdependent criteria.

Relational expressions are the foundational elements of nested conditionals. These expressions compare two values using relational operators such as greater than, less than, equal to, or not equal to. The result of a relational expression is a Boolean value that determines whether a specific branch of code executes. Some relational operators are outlined in the chart below:

Operator	**Meaning**
>	Greater than
<	Less than
==	Equal to
!=	Not equal to

To construct more complex decision-making logic, programmers often combine relational expressions using Boolean operators to form logical expressions.

(6.AP.2e) Logical expressions and, or, and not can be combined to compare two values and determine if a statement is true or false. The ability to predict the outcome of logical expressions is essential for understanding and debugging program behavior. Accurately interpreting how logical conditions evaluate in different scenarios allows programmers to determine whether specific branches of code will execute. Developing proficiency in constructing and analyzing logical expressions supports computational thinking and enhances a student's ability to write efficient, logically sound algorithms.

Concepts and Connections

Concepts

Computational thinking in programming involves applying logic and structured processes to solve problems through code. This includes designing decision trees, using Boolean, integer, and decimal variables, and creating programs that collect numeric input from users. It also requires writing nested conditional statements and evaluating logical expressions using Boolean operators (and, or, not), including those that combine relational conditions.

Connections

Within the grade level/course: At this grade level, students plan and implement algorithms incorporating sequencing, loops, variables, user input, conditional control structures, and functions using block-based or text-based programming tools. These computational skills are integral across various subjects, enhancing students' ability to organize, analyze, and interpret information systematically. (6.AP.2).
Vertical Progression: In Grade 5, students include nested conditional control structures (5.AP.2) In Grade 7, students continue practice implementing algorithms with sequencing, loops, variables, user input, conditional control structures and functions. (7.AP.2)

Across Content Alignment

Mathematics

- **6.PS.2** The student will represent the mean as a balance point and determine the effect on statistical measures when a data point is added, removed, or changed. The student will estimate, represent, solve, and justify solutions to single-step and multistep contextual problems using addition, subtraction, multiplication, and division with whole numbers.

Science

- **6.1** The student will demonstrate an understanding of scientific engineering practices by a) asking questions to determine relationships between independent and dependent variables, b) planning and carrying out investigations, c) interpreting, analyzing and evaluating

data, d) constructing and critiquing conclusions and explanations, e) developing and using models, f) obtaining, evaluating and communicating information. . *6.1 standard is integrated within science content and not taught in isolation.*

DIGITAL LEARNING INTEGRATION

- **6-8.ID** Students use a variety of technologies, including assistive technologies, within a design process to identify and solve problems by creating new, useful or imaginative solutions or iterations.
  A. In collaboration with an educator, students use appropriate technologies in a design process to generate ideas, create innovative products, or solve authentic problems.

Opportunities for Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

History and Social Science

- Students design a decision tree or program to model how laws or policies are influenced. By using conditional logic and Boolean operators, they simulate real-world scenarios such as citizen petitions or voting outcomes to explore how input data can affect government decisions.

Mathematics

- Students apply conditional control structures to design algorithms that solve real-world math word problems. By using logic to determine appropriate operations based on context (e.g., addition for "more than," subtraction for "less than"), they build programs that handle numeric input and apply nested conditions to perform accurate calculations.

Science

- Students write a program that gathers numeric data (such as temperature or measurements) at defined intervals. Using conditional and nested control structures, they build logic that responds dynamically to input (e.g., displaying warnings when temperature exceeds 30°C), reinforcing real-world applications of data-driven decision-making.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

1. Decompose Real-World Problems
2. Explore Common Features and Relationships to Extract Patterns
3. Apply Abstraction to Simplify, Represent, and Problem Solve
4. Apply Algorithmic Thinking to Problem Solve and Create
5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems
2. Plan and Design Artifacts
3. Create, Communicate and Document Solutions
4. Test and Optimize Artifacts

6.AP.3 The student will use the iterative design process to create, test, and debug programs containing sequencing, loops, variables, user inputs, nested conditional control structures, and two-way branching conditional control structures in a block-based programming tool.

a. Use accurate terminology to describe and explain the iterative design process.
b. Create and test programs that consist of sequencing, loops, variables, user inputs, nested conditional control structures, and two-way branching conditional control structures.
c. Trace and predict outcomes of programs.
d. Analyze and describe program results to assess validity of outcomes.
e. Revise and improve programs to resolve errors or produce desired outcomes.

Understanding the Standard

The iterative design process is a systematic method for developing and refining products, systems, or solutions through repeated cycles of design, evaluation, and revision. This process is often non-linear, allowing for continuous improvement and error resolution as new information and insights emerge. In programming, it serves as a fundamental development methodology, enabling students to engage in hands-on problem-solving through iterative cycles of designing, testing, debugging, and refining code. This approach reinforces the understanding that programming is not a one-time task, but an ongoing, dynamic process that evolves in response to feedback, results, and changing requirements.

[6.AP.3a] The iterative design process is a structured method for developing and improving code through repeated development cycles. It begins with designing an initial algorithm or program structure aligned to a defined problem or specification. Code is written to implement the solution using constructs such as loops, conditionals, variables, and functions. The program is then tested to verify expected behavior under standard and edge case scenarios. Errors are identified and corrected through debugging, addressing logic flaws, syntax issues, or runtime failures. Final steps involve refining the code to enhance efficiency, maintainability, or performance.

Each iteration provides an opportunity to assess the effectiveness of the solution, make corrections, and deepen understanding of both programming concepts and the problem being solved. The iterative process mirrors real-world software development practices and reinforces the idea that quality code is achieved through revision, evaluation, and continuous learning.

[6.AP.3b] Teachers should ensure that students write, test, and debug programs using the following constructs to support the development of logical reasoning, problem solving, and computational thinking.

- Sequence is the specific order in which instructions are executed in a program. The flow of control refers to this order, and incorrect sequencing can affect whether the program runs properly. Loops are used to repeat actions efficiently, while variables store and update data throughout the program.

- Loops are the repeated execution of a block of code until a certain condition is met (e.g., "for" loops or "while" loops). By repeating commands, programmers write fewer lines of code and reduce the chances of making errors.
- User input refers to data provided by the user of the program, such as entering text (e.g., a username) or numbers (e.g., selecting a quantity for purchase).
- Conditional control structures are structures (e.g., "if-else" statements) analyze variables and control whether certain commands are run based on conditions.
- Nested conditional structures involve placing one conditional statement inside another, supporting more complex decision-making when multiple criteria must be evaluated.
- Two-way branching conditional control structures allows the program to choose between two possible paths based on a condition.

[6.AP.3c] Tracing is the process of reviewing an algorithm or program step by step to observe how variable values change during execution. This involves analyzing control flow and logic structures to determine how data is processed at each stage. Tracing may be done manually or with debugging tools to identify variable states, output values, and execution paths. Reviewing variable changes supports verification of correctness and identification of inconsistencies in the program's behavior.

A recommended approach in tracing the flow of an algorithm or program includes the following:

1. Read the entire program to understand its overall structure and purpose.
2. Identify the variables used and note their initial values.
3. Start at the first line of code and move line by line in order the program runs.
4. Record the value of each variable as it changes after every line of execution.
5. Follow control structures like conditionals (e.g., *if/else*) and loops, and note which path or iteration is taken based on the conditions.
6. Predict the program's output based on the variable values and logic you've traced.
7. Compare traced output to the actual result when the program runs to check for accuracy and identify any logic or syntax errors.

[6.AP.3d] After a program executes, the output must be evaluated to determine whether it aligns with expected results. This evaluation involves verifying program correctness and identifying deviations such as incorrect variable updates, unmet conditions, or logic errors. When an algorithm does not produce the intended result, a systematic review process is used to identify and resolve faults. This includes tracing execution flow, analyzing each logical step, and comparing actual outputs to expected values. Tools such as debugging environments, print statements, or manual walkthroughs support error detection and correction.

Common modifications include correcting variable assignments, revising control structures such as loops or conditionals, and refining logic expressions. These changes are made to restore functional accuracy and align the algorithm's behavior with its original design specifications.

Following revisions, the algorithm is retested to verify that the issue has been resolved. This process reinforces the iterative nature of software development, where repeated testing, analysis, and refinement are used to improve performance and reliability.

- For example, in a game design project, students may analyze how a score variable changes based on player actions within a loop and evaluate how conditional statements impact the game's difficulty. By examining these results, students determine whether the scoring logic aligns with the intended game mechanics, verify that conditions trigger the correct responses, and make adjustments to improve gameplay balance. This process reinforces the importance of reviewing program behavior to ensure logical consistency and user engagement.

[6.AP.3e] Revising and improving an algorithm is a fundamental part of the debugging process, which involves identifying and correcting errors that prevent a program from functioning as intended. Debugging focuses on isolating logical or syntactical faults, commonly referred to as "bugs", within a program's code or structure.

Debugging is the process of identifying and correcting errors within a program. It involves tracing variable states, monitoring control flow, and analyzing outputs at specific execution points. Built-in debugging tools, breakpoints, and output statements are used to locate faults in logic, syntax, or runtime behavior.

Effective debugging isolates the cause of an error and supports targeted revisions. It contributes to functional accuracy, consistency with program specifications, and overall program stability. Debugging is a core component of iterative development and is repeated throughout the programming cycle to ensure the final solution operates as intended. Consider the following example checklist for debugging a program:

1. Syntax Errors
   - Are there any typos (spelling mistakes, incorrect keywords)?
   - Are parentheses, braces, and quotes properly placed?
   - Is there correct indentation or block structure?
2. Logic Errors
   - Does the program do what I expect? (i.e. correct math? right output?)
   - Are the if condition (if-statements) correct?
   - Are loops running too many or too few times?
3. Variables
   - Is the variable name spelled correctly?
   - Is the variable being used consistently throughout the code?
   - What is stored in the variable?
   - Is it the correct type of value?

Concepts and Connections

CONCEPTS

The iterative design process involves planning, developing, testing, and refining programs to improve functionality and correctness. It requires the use of programming constructs such as sequencing, loops, variables, conditionals, and user input, along with accurate terminology to explain each step. Tracing code, validating outputs, and debugging are essential for identifying errors and optimizing performance.

CONNECTIONS

Within the grade level/course: At this grade level, students work to collaborate on programs and assist each other with troubleshooting algorithms. They must be able to develop model representations. Students learn how to provide useful feedback to their peers when tracing and debugging code. (6.AP.3)

Vertical Progression: In Grade 5, students use accurate terminology to describe and explain the iterative design process, create programs that include two-way branching conditional control structures, and trace and predict the outcomes of programs (5.AP.3). Ing Grade 7, students use the iterative process to create, test and debug programs with a goal of identifying logic and syntax errors and resolving them to produce desired outcomes. (7.AP.3)

ACROSS CONTENT AREAS

Science

- **6.1** The student will demonstrate an understanding of scientific engineering practices by a) asking questions to determine relationships between independent and dependent variables, b) planning and carrying out investigations, c) interpreting, analyzing and evaluating data, d) constructing and critiquing conclusions and explanations, e) developing and using models, f) obtaining, evaluating and communicating information. . *6.1 standard is integrated within science content and not taught in isolation.*

DIGITAL LEARNING INTEGRATION

- **6-8.EL** Students leverage technologies, including assistive technologies, to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.
  C. Students seek feedback from people, including peers, teachers, staff familiar with assistive technologies, and functionalities embedded in technologies to make changes to improve and demonstrate their learning.

- **6-8.ID** Students use a variety of technologies, including assistive technologies, within a design process to identify and solve problems by creating new, useful or imaginative solutions or iterations.
  B. In collaboration with an educator, students select and use appropriate technologies to plan and manage a design process that identifies design constraints and trade-offs and weighs risks.

C. In collaboration with an educator, students use appropriate technologies in a cyclical design process to develop prototypes and demonstrate the use of setbacks as potential opportunities for improvement.

- **6-8.DC** Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act in ways that are safe, legal, and ethical.
  C. Students demonstrate and advocate for an understanding of intellectual property with both print and digital media—including copyright, permission, and fair use.
- **6-8.CC** Students communicate clearly and express themselves creatively for a variety of purposes using appropriate technologies (including assistive technologies), styles, formats, and digital media appropriate to their goals.
  A. Students select and use appropriate technologies to create, share, and communicate their work effectively, considering the audience.
  D. Students select and use appropriate technologies to design, publish, and present content that effectively convey their ideas, conclusions, and evidence for specific audiences.
- **6-8.GC** Students use appropriate technologies, including assistive technologies, to broaden their perspectives and enrich their learning by collaborating with others and working effectively in teams locally and globally.
  B. Students determine their role on a team based on their knowledge of content and technologies, as well as personal preference, and use appropriate technologies to track team progress toward a common goal.

Opportunities for Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students create interactive, branching stories using conditional logic, allowing users to make choices that affect narrative outcomes. Through an iterative design process, they write, debug, and refine their programs—incorporating media elements and peer feedback to ensure each decision leads to a coherent and engaging result.

Mathematics

- Students design a program to solve multi-step math word problems by using conditional control structures to select appropriate operations based on context. Through an iterative design process, they refine the program's logic and structure incorporating peer feedback and debugging techniques to ensure accuracy and adaptability across various problem types.

Science

- Students will explore the human digestive system by conceptualizing it as a machine with interdependent parts. Students will use abstraction to isolate essential components (organs, enzymes, functions) and build physical or digital models to demonstrate how materials move through each stage. By decomposing the system and focusing only on key processes, students will strengthen their understanding of biological functions and systems thinking.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

A. Fostering Collaborative Computing Practices:

1. Cultivate Relationships and Norms
2. Include Multiple Perspectives
3. Create and Accept Feedback

B. Fostering Computational Thinking Practices:

1. Decompose Relevant Problems
2. Explore Common Features and Relationships to Extract Patterns
3. Apply Abstraction to Simplify, Represent, and Problem Solve
4. Apply Algorithmic Thinking to Problem Solve and Create
5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data Sets

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-world Problems:
2. Plan and Design Artifacts
3. Create, Communicate and Document Solutions
4. Test and Optimize Artifacts

6.AP.4 The student will demonstrate proper attribution when incorporating ideas and works of others.

a. Identify and give proper attribution of information and assets from the Internet and other sources.

Understanding the Standard

Programs may integrate existing algorithms, code segments, libraries, and media assets to optimize development time and improve functionality. Incorporating pre-existing components allows developers to utilize proven solutions for common computational tasks, reducing redundancy and supporting modular, scalable design. These resources may include pre-written functions, reusable modules, or third-party packages that perform specific operations such as sorting, encryption, or user interface rendering.

Code libraries provide structured collections of functions and classes that abstract complex processes into simpler calls. For example, libraries for data visualization, machine learning, or graphics rendering enable developers to implement advanced features without writing low-level code. Code snippets may also be reused to implement frequently used patterns or algorithms, increasing consistency and reducing the likelihood of introducing errors.

Media assets, such as images, audio files, or animation sequences, are commonly imported to enhance user interface or user experience components. The integration of these digital assets into programs requires appropriate referencing, licensing compliance, and optimization for system performance.

- For example, in game development, external code may be integrated to simulate realistic jump mechanics, and background images licensed under Creative Commons can be imported to enhance visual design. Creative Commons licenses define how digital content may be used, shared, or modified while respecting intellectual property rights.

External components incorporated into a program must comply with licensing requirements. These licenses define how a component may be used, modified, and distributed within a project.

[6.AP.4a] Proper attribution is a legal and ethical requirement that acknowledges the original creator of an external resource. This may involve including license text in project documentation, referencing the author in code comments, or displaying credit within the user interface. Attribution ensures compliance with usage terms and supports transparency in the development process.

Key Steps to Proper Attribution:

- Identify the source: Ensure students know where their materials come from, this includes the website, author, or source URL.
- Use correct citation format: Just like citing a book or article, students should include key details such as the author's name, the title of the work, and the URL.

- Place the attribution properly: In coding, attribution often happens in comments within the code or in a designated “Credits” section of the program.

Attribution may be provided as comments within the code, in a dedicated credits section of the user interface, or in accompanying documentation such as a README file. The placement of attribution should ensure it is clearly visible and accessible to others reviewing, using, or modifying the program. Proper attribution typically includes the original creator’s name, the title or description of the asset, a link to the source, and the applicable license (e.g., Creative Commons).

Concepts and Connections

Concepts

Proper attribution of information and digital assets requires identifying the original source and accurately citing content, including text, images, audio, video, and other media. This practice ensures compliance with copyright laws, respects intellectual property rights, and supports ethical use of online and external resources.

Connections

Within the grade level/course: At this grade level, students may use portions of code which have already been created and must learn to provide proper attribution to author and source while following legal and ethical guidelines in doing so. Highlighting the balance between utilizing existing resources to enhance programming efficiency and creativity, while also emphasizing the importance of respecting intellectual property rights and ethical practices is imperative to student development. (6.AP.4)
Vertical Progression: In Grade 5,In Grade 7, Students apply proper methods of attribution when incorporating information from the internet or other sources into their programs. (7.AP.4)

Across Content Areas

English

- **6.R.1** Evaluation and Synthesis of Information
- **6.R.1f** Give credit for information quoted or paraphrased using standard citations (e.g., author, article title, webpage, and publication date).
- **6.R.1g** Demonstrate ethical and responsible use of all sources, including the Internet, Artificial Intelligence (AI), and new technologies as they develop.

History and Social Science

- **CE.13** The student will apply social science skills to understand the role of government in the United States economy by e) describing how governments regulate to protect consumers, labor, the environment, competition in the marketplace, and property rights.

DIGITAL LEARNING INTEGRATION

- **6-8.DC** Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act in ways that are safe, legal, and ethical.
  C. Students demonstrate and advocate for an understanding of intellectual property with both print and digital media—including copyright, permission, and fair use.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students research a historical figure or scientific discovery using multiple sources and incorporate media elements such as images, videos, and quotes. Students apply proper attribution to accurately attribute all referenced materials in a written report or multimedia presentation.

History and Social Science

- Students create a multimedia presentation or video that explores a historical event, enriching their work with digital assets such as images, videos, and music. Students apply proper attribution to cite all media sources, demonstrating digital and ethical responsibility.

Mathematics

- Students design interactive math games such as quizzes, puzzles, or challenges to help others practice key concepts like ratios, rates, and percentages. As part of the development process, they incorporate pre-made digital assets from online sources and apply proper attribution practices to ensure ethical use of all external materials.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices.
2. Safeguard Well-Being of Self and Others.
3. Evaluate Resources and Recognize Contributions.

Computing Systems (CSY)

6.CSY.1 The student will define and explain application software and operating systems of a computing device within a computing system.

a. Define and describe the functions of an operating system and application software.
b. List advantages and limitations of application software and operating systems based on the needs of the user.

Understanding the Standard

[6.CSY.1a] A computing system is a combination of hardware and software that works together to perform input, processing, storage, and output functions. A computing system relies on two primary types of software: system software, which manages hardware resources and system operations, and application software, which enables users to perform specific tasks such as word processing or web browsing.

- An operating system (OS) serves as the foundational software layer that manages the overall operation of a computer system. It controls and coordinates hardware components, allocates system resources (such as memory, processing power, and storage), and provides the user interface through which users interact with the system. The OS also facilitates the execution of application programs and ensures stable, efficient system performance.
- Application software refers to programs designed to perform specific user-oriented tasks. This category includes general productivity tools such as word processors, spreadsheets, and presentation software as well as domain-specific applications used in fields like engineering, finance, healthcare, and education.

[6.CSY.1b] The selection of an operating system and application software is determined by specific user requirements, technical specifications, and system performance goals. Core evaluation criteria include interface functionality, compatibility with existing hardware and software, efficiency in resource utilization, and alignment with task-specific needs. Operating systems differ in kernel architecture, file system structure, and supported application ecosystems, all of which influence system behavior and capabilities.

Security features such as user account control, encryption protocols, and access permissions are critical in environments that handle sensitive data. Users with specialized workflows such as those in digital content creation or data analysis may require operating systems that support advanced graphics processing units (GPUs), high-throughput memory allocation, or compatibility with industry-standard applications. Enterprise environments may prioritize features such as virtualization support, centralized administration, and remote access management.

Application software must also be selected based on licensing structure (proprietary vs. open source), software update mechanisms, and support channels. Integration with cloud-based services and cross-platform synchronization may be necessary in collaborative or mobile work settings. Aligning the operating system with the application layer ensures that computing processes are performed reliably, securely, and in accordance with user-defined performance benchmarks.

Concepts and Connections

CONCEPTS

Operating systems control hardware resources and support the execution of application software, which is designed to perform specific user tasks. Comparing the advantages and limitations of each involves analyzing factors such as system compatibility, performance efficiency, available features, and cost relative to user requirements.

CONNECTIONS

Within the grade level/course: At this grade level, students explain application software and operating systems of a computing device within a computing system. (6.CSY.1)

Vertical Progression: In Grade 5, students explore how computing systems exchange information through networks and digital communication protocols. (5.CSY.1). In Grade 7, students design projects that collect and exchange data and evaluate the effectiveness of hardware and software components. (7.CSY.1)

DIGITAL LEARNING INTEGRATION

- **6-8.DC** Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act in ways that are safe, legal, and ethical.
  D. Students demonstrate an understanding of what personal data is, how data collection technologies work, tradeoffs of sharing personal data, and best practices for keeping it private and secure.
- **6-8.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  B. Students find or organize data and use appropriate technologies to interpret, analyze, and represent data to construct models, predict outcomes, solve problems, and make evidence-based decisions.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

History and Social Science

- Students investigate how legal institutions utilize software and operating systems to support tasks like case management, evidence tracking, and document filing. Through their research, they examine the specific digital tools employed in legal workflows and analyze the role of technology in streamlining judicial and administrative processes.

Mathematics

- Students critically examine various software tools used to solve mathematical problems such as spreadsheets and graphing calculators by comparing their technical features, computational capacities, and practical limitations. Through contextualized problem-solving (e.g., tax calculations or data analysis), students differentiate between application software and operating systems, evaluating how each platform supports mathematical modeling, formula construction, and decision-making in authentic scenarios.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

C. Fostering Iterative Design Practices:

3. Create, Communicate, and Document Solutions

D. Fostering Digital Literacy Practices:

3. Evaluate Resources and Recognize Contributions

6.CSY.2 The student will identify and explain hardware, software, and connectivity problems and troubleshooting solutions.

a. Identify and explain hardware, software, and connectivity problems and solutions with accurate terminology.
b. Identify resources for troubleshooting hardware, software, and connectivity-related problems.

Understanding the Standard

[6.CSY.2a] Troubleshooting computing systems involves the systematic process of identifying, diagnosing, and resolving issues related to hardware, software, or connectivity. Accurate classification of the problem domain allows for targeted resolution. Hardware issues may include non-responsive peripherals, overheating, or device failure. Software issues often manifest as application crashes, error messages, or inconsistent behavior caused by corrupted files or incompatible configurations. Connectivity issues typically involve slow data transfer rates, failure to access network resources, or total loss of internet access.

Standard troubleshooting procedures are applicable across most computing systems. These may include verifying physical connections, ensuring power delivery, or replacing suspected faulty components with known working substitutes. Restarting a system is a frequent and effective initial step that resets processes and clears temporary memory. For cloud-based environments, users often troubleshoot connectivity by confirming network settings, signal strength, or authentication credentials.

Effective troubleshooting requires isolating the root cause. This can involve reviewing device logs, applying software patches, checking for driver conflicts, or modifying network configurations. Troubleshooting processes often follow structured logic, similar to conditional branching in programming. For example, protocols might apply a sequence such as: if the system boots, then check the output display; if the system does not boot, then evaluate power supply integrity. This structured, decision-based approach supports efficient identification and resolution of computing system failures.

Example: A systematic process to diagnosing hardware issues may consist of the following steps:

Verify power and device activation	Inspect physical connections and peripherals	Check basic settings	Listen for diagnostic sounds
• Ensure the device is plugged into a functioning power source. • Confirm that the power switch is turned on.	• Ensure all cables are securely connected. • Disconnect unnecessary external devices to isolate the issue.	• Perform a complete system reboot to reset hardware components and resolve temporary issues.	• Observe any startup beeps, fan noise, or drive activity that could indicate hardware function or failure.

• Check that the display or monitor is powered and active.	• For portable devices, confirm battery charge or power adapter functionality.		

Example: A systemic process for diagnosing and resolving software-related issues may include the following steps:

Identify the problem	Restart the application	Check for software updates	Test software on another device
• Determine the specific software or application experiencing issues. • Record any visible error messages, codes, or unexpected behavior to support further diagnosis.	• Close and relaunch the software to clear temporary glitches or memory conflicts.	• Ensure the application and operating system are updated to the latest version to resolve known bugs or compatibility issues.	• Run the same application on a different device to determine whether the issue is device-specific or related to the software itself.

Example: A systematic approach to diagnosing network or internet connectivity issues may include the following steps:

Verify Wi-Fi connection status	Check Airplane Mode and wireless settings	Test access to online content	Disconnect and reconnect to the network
• Ensure the device is connected to the intended Wi-Fi network. • Reconnect to the network if the connection is lost or unstable.	• Confirm that Airplane Mode is turned off and that Wi-Fi is enabled. • On some devices, ensure mobile data is enabled if Wi-Fi is not in use.	• Try loading multiple websites or using different internet-based apps to determine if the issue is isolated to one service or more widespread.	• Manually disconnect from the Wi-Fi network and reconnect to initiate a fresh session, which can resolve authentication or IP assignment issues.

Accurate technical terminology is essential in troubleshooting computing systems. Using domain-specific vocabulary ensures clear communication when diagnosing issues and applying solutions. Precise terminology supports the correct interpretation of system documentation, technical manuals, and diagnostic tools.

In collaborative IT environments, consistent terminology helps maintain coherence when reporting problems, seeking support, or documenting resolutions. It also enables effective communication with support personnel, manufacturers, or technical communities. Mastery of this vocabulary improves efficiency in problem-solving and contributes to maintaining system reliability.

[6.CSY.2b] Troubleshooting hardware, software, and connectivity issues involves identifying the source of a problem and applying targeted solutions. Effective troubleshooting relies on accurate resources such as user manuals, system diagnostics, manufacturer websites, and IT support personnel. Built-in tools like Windows Troubleshooter, macOS Disk Utility, and network diagnostics assist in isolating system-level faults. Reviewing system logs and error messages also supports identifying specific causes of failure.

Manufacturer support sites provide essential documentation, drivers, and direct technical support. Online help forums and technical communities share peer-reviewed solutions and common fixes. Educational platforms deliver tutorials for step-by-step troubleshooting processes. School or organizational IT support personnel offer direct assistance for system maintenance. Together, these tools and resources contribute to effective digital problem-solving and consistent system performance.

Concepts and Connections

CONCEPTS

Hardware, software, and connectivity issues must be clearly identified and described using accurate technical terminology, along with appropriate diagnostic and corrective actions. Effective troubleshooting includes the use of reliable resources such as system documentation, diagnostic utilities, error logs, and technical support services to resolve problems efficiently

CONNECTIONS

Within the grade level/course: At this grade level, students accurately identify hardware, software, and connectivity problems when troubleshooting. (6.CSY.2)
Vertical Progression: In Grade 5, students use computational thinking strategies when identifying a problem and following protocols to fix the problem. (5.CSY.3). In Grade 7, students apply computational thinking and troubleshooting skills to systemically process and resolve

problems. They compile and record all successful methods used to identify common hardware, software and connectivity related problems. (7.CSY.2).

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

Mathematics

- Students use spreadsheet software to solve multi-step math problems involving topics like percentages, ratios, and basic budgeting. During the process, they encounter and resolve realistic digital challenges such as broken formulas, missing cells, or formatting errors by applying mathematical reasoning and collaborating to troubleshoot. Students document the steps taken to identify and fix each issue,

Science

- Students investigate how scientific tools such as sensors, microscopes, or weather stations operate through the interplay of hardware and software and simulate troubleshooting scenarios to address common data collection failures. Through guided experimentation, they identify issues including malfunctioning devices, connectivity disruptions, and software errors, using accurate technical vocabulary to articulate diagnoses. Students research external resources (e.g., digital manuals, help guides, forums) to propose and evaluate solutions, documenting their reasoning and reflecting on how effective data collection depends on reliable system performance.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

1. Decompose Real-World Problems
4. Apply Algorithmic Thinking to Problem Solve and Create

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems
3. Create, Communicate and Document Solutions
4. Test and Optimize Artifacts

6.CSY.3 The student will identify and describe Artificial Intelligence (AI).

a. Define artificial intelligence and identify the characteristics of artificial intelligence.
b. Describe how AI technologies generate information or automate decision and how people interact with AI technologies.
c. Define algorithmic bias and explain its consequences on AI technologies and systems.

Understanding the Standard

[6.CSY.3a] Artificial intelligence (AI) is a branch of computer science focused on creating algorithms and models that enable machines to perform tasks that typically require human intelligence. These tasks include learning from data, recognizing patterns, understanding language, solving problems, and making decisions. AI methods are embedded in a wide range of systems from standalone software applications to complex hardware environments, allowing them to operate autonomously, adapt over time, and support real-world functionality across disciplines and industries.

Artificial intelligence (AI) is focused on designing intelligent agents, software systems that perceive their environment, make decisions, and act to achieve goals. An intelligent agent is software that has been designed to use artificial intelligence and machine learning to perform tasks without human intervention. AI systems are designed to mimic human intelligence, enabling them to perform tasks that would typically require human cognitive abilities.

Artificial intelligence (AI) systems rely on foundational capabilities to perform assigned tasks across a different digital environment. These capabilities enable intelligent agents to interpret data, adapt to changing conditions, and make decisions based on goals or constraints. Core functions include perception, representation and reasoning, learning from data, and interacting with humans in natural ways:

- Perception refers to how computers interpret the physical world through sensors. It involves the process of extracting meaningful information from raw sensory input such as images, sounds, or environmental signals.
- Representation and reasoning describe how intelligent agents maintain internal models of the world using data structures. These representations enable reasoning algorithms to derive new conclusions from existing knowledge, supporting decision-making and problem-solving.
- Learning occurs when computers analyze large datasets to identify patterns. This process, known as machine learning, depends on the availability of training data—often curated or labeled by humans—to improve the system's accuracy and performance over time.
- Natural interaction involves the ability of intelligent systems to communicate and engage with humans in ways that feel intuitive. This requires knowledge of language, emotion, facial expressions, and an understanding of cultural and social norms. Due to the complexity of human communication and limited general reasoning capabilities, enabling natural interaction remains a challenging area of artificial intelligence.

[6.CSY.3b] Artificial intelligence (AI) technologies use algorithms and statistical models to analyze structured and unstructured data at scale. These systems apply machine learning techniques to identify patterns, classify inputs, and make data-driven predictions or decisions without being explicitly programmed for every scenario. Supervised, unsupervised, and reinforcement learning approaches allow AI systems to generate insights or optimize outcomes based on historical or real-time data.

Human interaction with AI technologies is typically facilitated through interfaces powered by natural language processing (NLP), computer vision, or speech recognition systems. These components enable AI to interpret spoken commands, understand written text, or analyze images and videos.

AI technologies generate output based on data used during training. Human interaction with AI occurs through systems such as conversational agents, recommendation engines, and autonomous systems. Models trained on biased, inaccurate, or limited data may produce outputs that are unreliable or lack transparency. Evaluation protocols and ethical oversight define practices for responsible AI design and implementation.

[6.CSY.3c] Algorithmic bias occurs when the design of an algorithm or the data used to train it leads to inaccurate outputs. This bias originates from imbalanced or non-representative training data, flawed model design, or embedded human assumptions encoded into the system.

Concepts and Connections

CONCEPTS

Artificial intelligence (AI) refers to the capability of machines to perform tasks that typically require human cognitive abilities, such as learning, reasoning, and problem-solving. AI technologies generate information and automate decision-making through data processing and algorithms, while human interaction occurs through tools like virtual assistants, recommendation systems, and autonomous technologies. Algorithmic bias is a systematic error in AI systems caused by biased data or flawed design, resulting inaccurate outcomes.

CONNECTIONS

Within the grade level/course: At this grade level, students identify and describe artificial intelligence (AI) systems by examining how these technologies utilize algorithms and data to detect patterns, generate predictions, and inform decision-making processes. (6.CSY.3)
Vertical Progression: In Grade 7, students will explain role of neural networks in machine learning and artificial intelligence. (7. DA.2).

ACROSS CONTENT AREAS

English

- **6.DSR** The student will build knowledge and comprehension skills from reading a range of challenging, content-rich texts. This includes fluently reading and gathering evidence from grade-level complex texts, reading widely on topics to gain purposeful knowledge and vocabulary, and using reading strategies when comprehension breaks down.

(Note: text must be domain specific and applied)

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students research a real-world application of artificial intelligence such as facial recognition, autonomous vehicles, or algorithmic decision-making in hiring and analyze its benefits, limitations, and ethical implications. To build background knowledge and strengthen comprehension, students will engage with a range of challenging, content-rich texts that include journalistic articles, academic papers, opinion pieces, and primary sources.

History and Social Science

- Students research a real-world example of artificial intelligence implemented within government systems such as AI-assisted public transportation or automated decision-making in policy enforcement. Students define artificial intelligence and explain how it functions within these civic structures, grounding their analysis in technical, historical, and ethical frameworks.

Mathematics

- Students will explore how artificial intelligence is used to analyze sports statistics, predict outcomes, and recommend products online by applying math concepts such as data analysis, probability, and patterns.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems
2. Create, Communicate and Document Solutions

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices.
2. Safeguard Well-Being of Self and Others.

3. Evaluate Resources and Recognize Contributions.

Cybersecurity (CYB)

6.CYB.1 The student will evaluate the risks and benefits associated with sharing personal and public resources or artifacts.

a. Identify and explain the difference between personal and public information.
b. Discuss the consequences of sharing personal and confidential information online.
c. Evaluate risks and benefits associated with sharing information online.

Understanding the Standard

[6.CYB.1a] Information shared or accessed in digital environments can fall into different categories, with varying expectations for privacy, accessibility, and protection.

- Personal information consists of data that can directly or indirectly be used to identify an individual. This includes identifiers such as full name, home address, phone number, Social Security number, and date of birth. Such information is considered sensitive because it can be misused for identity theft, fraud, or other unauthorized activities. Personal information requires protection under privacy regulations and should only be accessed or shared with appropriate consent or legal justification.

- Public information consists of data that is legally available to the public and does not pose a risk to individual privacy when disclosed. Examples include government press releases, published research, meeting minutes, and open-source datasets. This type of information is typically accessible without restriction and is intended for broad distribution and transparency.

The key distinction between personal and private information is the level of confidentiality and required safeguards. Personal information requires limited access and secure handling to prevent misuse, while public information is designed for open access and general use.

[6.CYB.1b] Personal information is shared online through account registrations, online forms, social media platforms, and blogs. This includes names, contact information, identifiers, and other data linked to an individual. Public-facing content, such as social posts and comments, often contains voluntarily disclosed personal details. Sharing this information can expose individuals to identity theft, fraud, cyberbullying, and unauthorized profiling. Personally identifiable information may be collected, stored, and associated with user behavior across multiple digital platforms.

[6.CYB.1c] Online platforms enable users to share personal information to access services such as digital banking, e-commerce, healthcare portals, and government systems. Common interactions include submitting forms, processing payments, and managing accounts. These exchanges offer efficiency, real-time access to services, personalized user experiences, and streamlined workflows for daily tasks. Social platforms also support digital communication, information sharing, and participation in virtual communities.

Sharing personal information online also introduces risk. Data may be collected, stored, or transferred without explicit user consent. Common threats include unauthorized access, identity theft, phishing, cyberbullying, data profiling, and the commodification of personal information. Once data is exposed, it may be used for malicious purposes or retained in third-party systems beyond the user's control, increasing the likelihood of reputational, emotional, or financial harm.

Concepts and Connections

Concepts

Personal information, or personally identifiable information (PII), includes data such as name, address, and login credentials that can identify an individual, while public information is intended for open access and poses minimal privacy risk. Sharing personal or confidential information online can lead to identity theft, financial fraud, and unauthorized access. Evaluating what to share requires understanding the difference between personal and public information and considering potential risks to privacy and security

Connections

Within the grade level/course: At this grade level, students evaluate the risks and benefits associated with sharing personal and public resources or artifacts. (6.CYB.1)
Vertical Progression: In Grade 5, students identify and explain ways to limit unauthorized access on computing devices.(5.CYB.1). In Grade 7, students differentiate physical and digital security measures that protect electronic information. (7.CYB.1).

Across Content Areas

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students define key terms related to digital privacy (e.g., personal data, cyberbullying, digital footprint) and analyze the risks and benefits of sharing personal information online using content-rich, age-appropriate texts.They will then write a letter to a younger student explaining the importance of keeping personal information safe, providing examples of both risks and benefits of sharing information online. The essay will also include advice on how to protect privacy and the possible outcomes of sharing too much information.

History and Social Science

- Students analyze federal and state regulations governing online data sharing, including the Children's Online Privacy Protection Act (COPPA), and evaluate the implications of those laws for digital citizenship. Drawing on informational texts and structured discussion, students will compare the risks and benefits of sharing personal information online, considering ethical, social, and safety dimensions. As a culminating activity, students will design and present a digital citizenship poster

Mathematics

- Students define how personal data is collected through apps, surveys, and social media, and classify different types of user-generated data (e.g., location, purchases, posts). Using sample datasets or classroom simulations, students will calculate ratios and percentages that reflect how much data is typically shared by different user profiles.

Science

- Students identify how internet activity and digital storage contribute to environmental challenges such as high energy consumption and electronic waste. Using curated data or teacher-provided resources, students will calculate estimates of energy use or e-waste produced by common digital behaviors (e.g., streaming, cloud storage, device upgrades).

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate real-world problems

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
2. Safeguard Well-Being of Self-Others

6.CYB.2 The student will investigate various usage agreements designed to protect individuals.

a. Identify laws governing privacy with computing devices and emerging technologies.
b. Investigate and describe common components of usage agreements.
c. Identify user and company protections in a usage agreement.

Understanding the Standard

[6.CYB.2a] Computing devices and emerging technologies collect, store, and transmit personal information during daily use. Legal frameworks define how this data must be managed to protect privacy and security. Usage agreements are binding contracts that specify the terms under which users may access digital services, software, and connected devices. These agreements address conditions for data collection, acceptable use, and liability and are required during software installation, online account registration, or access to public networks.

Federal and state laws govern the handling of personal information in computing environments. The Privacy Act of 1974 restricts how U.S. federal agencies collect, use, and disclose personally identifiable information, granting individuals the right to review and amend their records. The Children's Online Privacy Protection Act (COPPA) establishes data protection standards for online services targeting children under 13, including obtaining parental consent before collecting personal data. The Electronic Signatures in Global and National Commerce Act (E-SIGN Act) authorizes the use of electronic records and signatures in legal agreements, ensuring that digital contracts are enforceable.

State-level legislation, such as the Virginia Consumer Data Protection Act (VCDPA), expands consumer rights over personal data. It requires organizations to provide transparency about data practices and allows individuals to access, correct, or delete their personal information. These laws define the legal responsibilities of data controllers and processors in safeguarding information within digital systems.

[6.CYB.2b] Usage agreements, also known as terms of service or end-user license agreements, are legal documents that define the conditions under which users may access and use software, applications, or digital platforms. These agreements outline the rights and responsibilities of both the user and the provider.

Key components of usage agreements include user conduct policies that define permitted and prohibited actions, such as altering software or misusing services. Data privacy and security provisions identify how personal information is collected, stored, and used, and reference compliance with relevant data protection laws. Intellectual property clauses clarify ownership of software, content, and branding. Additional sections may include limitations of liability, service termination conditions, and procedures for dispute resolution. These agreements establish legal boundaries and operational expectations for digital products and services.

[6.CYB.2c] User agreements define the legal terms under which individuals and organizations access and use digital services. These agreements specify the rights, responsibilities, and limitations for both users and providers. Reviewing these agreements allows stakeholders to understand how data will be managed and what protections or restrictions apply.

User protections often include terms related to data privacy, specifying how personal information is collected, stored, processed, and shared. Agreements may outline content ownership, user rights related to data deletion or opting out of specific practices, and limitations of liability in the event of a data breach or system failure. Dispute resolution procedures and customer support protocols are also typically defined.

Company protections are designed to limit liability and maintain control over platform use. Clauses may reserve the right to change or terminate services, prohibit misuse of tools, enforce acceptable use policies, and assert intellectual property rights over content and software. These provisions support legal enforcement and operational flexibility for providers.

Concepts and Connections

Concepts

Privacy laws define how personal data must be collected, stored, and shared when using computing devices and emerging technologies. Understanding usage agreements requires analyzing key components such as terms of service, data collection practices, and user responsibilities. These agreements specify protections for users and organizations, including data security requirements, acceptable use policies, and legal limitations on data use and access.

Connections

Within the grade level/course: At this grade level, students examine user agreements to understand the rights and responsibilities associated with digital services. (6,CYB.2)
Vertical Progression: In Grade 5, students explain how cybersecurity policies and laws are designed to protect individuals. (5.CYB.2).

Across Content Areas

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students identify and interpret key language features used in real-world terms of service agreements for popular digital platforms and analyze whether the vocabulary and sentence structure are accessible to general users, particularly younger audiences.

History and Social Science

- Students research key privacy laws such as COPPA or FERPA and explain how they protect users' personal data and digital rights. Students will analyze the language and structure of usage agreements (e.g., terms of service, privacy policies) to evaluate how clearly users are informed about their protections and responsibilities.

Mathematics

- Students collect and classify statistics on data breaches and cybersecurity incidents across sectors such as social media, e-commerce, and gaming. Students calculate and compare percentages reflecting the frequency of personal data compromise with the inclusion of protective clauses in usage agreements.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices

Data and Analysis (DA)

6.DA.1 The student will utilize computational tools to collect and organize data.

a. Select and use appropriate computational tools to collect data.
b. Organize data to make it easier to understand and use.
c. Clean data to remove and correct errors.
d. Analyze data sources for accuracy and reliability.

Understanding the Standard

Computational tools play a critical role in supporting students as they visualize, analyze, and interpret data. These tools help identify patterns, detect trends, and make informed predictions based on large or complex data sets. Examples include spreadsheets, data visualization platforms, and coding environments each designed to process data and present it in accessible, meaningful formats. When integrated into the development of computational models, these tools allow students to simulate real-world systems by adjusting variables, observing outcomes, and evaluating hypotheses. This hands-on approach deepens students' conceptual understanding and supports evidence-based reasoning.

[6.DA.1a] Computational tools support the visualization, analysis, and interpretation of data. These tools include software applications such as spreadsheets, data visualization platforms, and coding environments that organize, process, and display data in interpretable formats. Computational tools also support the creation of models that represent real-world systems. By adjusting variables and observing outcomes, users can examine patterns, test ideas, and evaluate system behavior. Selecting appropriate computational tools depends on the type of data involved, the intended analysis, and the desired output format. Different tools offer different functionalities, making tool selection a critical step in effective data work.

[6.DA.1b] Organizing data into consistent structures makes the information easier to understand and use. This includes sorting values in a defined order, grouping related data into categories, labeling fields with clear identifiers, and aligning data types such as text, numbers, and dates. Structured data reduces ambiguity and ensures compatibility with computational tools. Well-organized data supports filtering, aggregation, and comparison across multiple dimensions. It also improves the accuracy of visualizations and enables tools to process data efficiently for analysis, modeling, and decision-making.

- For example, in a dataset tracking school attendance, organizing the data might involve sorting records by date, grouping students by grade level, using consistent labels like “Present” and “Absent,” and formatting all dates as MM/DD/YYYY.

[6.DA.1c] Cleaning data involves identifying and addressing errors, inconsistencies, or irrelevant information within a dataset. This step ensures the quality and reliability of data before analysis. Common tasks include removing duplicate records, correcting misspelled entries, standardizing formats, and filling in or removing missing values. Cleaning also includes checking for outliers or invalid data points that may

distort results. As part of the data cycle, cleaning occurs after data collection and before analysis, serving as a critical step to prepare data for accurate interpretation and effective visualization.

- For example, in a dataset of survey responses, cleaning may involve correcting misspelled words like “tomorow” to “tomorrow,” removing duplicate entries from the same participant, and converting all date entries to the same format (e.g., MM/DD/YYYY).

Clean data allows tools to sort, analyze, and visualize information correctly without distortion.

[6.DA.1d] Analyzing data sources for accuracy and reliability means checking the origin, quality, and context of the data. This includes verifying the source, confirming the data reflects current and complete information, and assessing whether the collection methods follow consistent procedures. Data must come from a trusted source, match the intended use, and avoid bias. This step ensures the data supports valid conclusions during analysis.

- For example, a dataset on local temperatures collected from a national weather agency is more reliable than one gathered from an unverified online post, because the agency uses standardized tools and methods to report accurate measurements.

Concepts and Connections

Concepts

Data collection involves using appropriate computational tools to gather accurate and relevant information. The cleaning process removes errors, duplicates, and inconsistencies to improve data quality. Once collected and cleaned, the data is organized into structured formats to support clarity, analysis, and usability.

Connections

Within this grade level: At this grade level, students utilize computational tools to collect data, organize data, clean datasets, and evaluate data sources for reliability.
Vertical progression: In Grade 5,students identify accurate ways to collect data, organize data based on patterns, and compare and contrast the various data elements to solve a problem or investigate a topic (5.DA.1). In Grade 7, students will utilize computational tools to visualize and evaluate data to draw conclusions and make predictions.(7.DA.1).

Across Content Areas

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data

sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Mathematics

- **6.PS.1** The student will apply the data cycle (formulate questions; collect or acquire data; organize and represent data; and analyze data and communicate results) with a focus on circle graphs.
- **6.PS.2** The student will represent the mean as a balance point and determine the effect on statistical measures when a data point is added, removed, or changed.

DIGITAL LEARNING INTEGRATION

- **6-8.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  B. Students find or organize data and use appropriate technologies to interpret, analyze, and represent data to construct models, predict outcomes, solve problems, and make evidence-based decisions.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

Mathematics

- Students collect and record personal or classroom data related to a math-based inquiry, such as average hours of sleep, steps per day, or screen time. Using spreadsheets or data analysis tools, students will organize, calculate, and visualize numerical patterns using grade-level math skills including averages, percentages, and basic graphs.

Science

- Students conduct a hands-on science experiment (e.g., measuring plant growth based on different watering levels) and collect observational data using spreadsheets or online data tracking tools. Students organize, clean, and format the dataset to ensure accuracy and usability for analysis.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

6.DA.2 The student will utilize computational tools to visualize and evaluate data.

a. Identify different types of visual representations of data.
b. Compare various visual representations and identify when each should be used.
c. Create charts, graphs, models, and simulations to visualize data.
d. Describe and synthesize information from a visual representation of data.

Understanding the Standard

Computational tools process raw data into formats that reveal patterns, trends, and relationships. These tools include charts, graphs, and dashboards that support comparison, measurement, and interpretation. Visual and evaluation tools help users examine data clearly and draw conclusions based on evidence.

[6.DA.2a] Data visualization refers to the use of visual formats to represent data in a clear and structured way. Charts, graphs, and other visual models help reveal patterns, trends, and relationships that may not be obvious in raw or text-based data. Visualizing data supports interpretation by placing information in a context that allows for faster recognition and comparison.

[6.DA.2bc] Visualization models use different formats to represent data clearly and support interpretation. Common types include:

- Circle graphs show percentages or parts of a whole, such as demographic composition or budget distribution.
- Bar graphs compare categorical data, including sales by product type or population by age group.
- Line plots show trends over time, such as changes in temperature or monthly revenue.
- Pictographs represent data using images or symbols and often appear in educational contexts to show preferences or survey results.
- Heat maps show the intensity or concentration of values across a geographic area, such as population density or crime levels.
- Geospatial maps display data tied to specific locations, including disease spread or the distribution of resources.

[6.DA.2d] Software applications support data visualization by organizing and displaying large datasets in clear, interpretable formats. These tools allow for rapid disaggregation, breaking down data by variables such as time, category, or demographic group. A model is a simplified, rule-based representation of a real-world system or process. Simulations apply the model to explore how a system behaves under changing conditions, often producing multiple outcomes for analysis. Software tools can generate both the model and simulation outputs, enabling users to test scenarios, identify trends, and compare results. The data produced from modeling and simulation supports forecasting, evaluation, and decision-making across domains such as science, economics, and public policy.

Concepts and Connections

CONCEPTS

Data visualization presents information using visual formats such as charts, graphs, models, and simulations. It involves selecting and comparing formats based on the structure, type, and intended use of the data. Effective visualizations simplify complex information, highlight patterns and relationships, and support data interpretation, analysis, and informed decision-making.

CONNECTIONS

Within this grade level: At this grade level, students select and use appropriate data visualizations to reveal patterns and insights that raw data may conceal. By exploring software tools, modeling, and simulation, they gain the ability to analyze systems and communicate findings effectively. (6.DA.2)

Vertical progression: In Grade 5, students analyze and use their data to create charts, graphs, and models so that they can make predictions, draw conclusions, and propose solutions to problems or questions based upon the data analysis (5.DA.2). In Grade 7, students will utilize computational tools to visualize and evaluate data to draw conclusions and make predictions. (7.DA.1).

ACROSS CONTENT AREAS

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Mathematics

- **6.PS.1** The student will apply the data cycle (formulate questions; collect or acquire data; organize and represent data; and analyze data and communicate results) with a focus on circle graphs.

- **6.PS.2** The student will represent the mean as a balance point and determine the effect on statistical measures when a data point is added, removed, or changed.

DIGITAL LEARNING INTEGRATION

- **6-8.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  B. Students find or organize data and use appropriate technologies to interpret, analyze, and represent data to construct models, predict outcomes, solve problems, and make evidence-based decisions.
- **6-8.CC** Students communicate clearly and express themselves creatively for a variety of purposes using appropriate technologies (including assistive technologies), styles, formats, and digital media appropriate to their goals.
  C. Students communicate complex ideas clearly using appropriate technologies to convey the concepts orally, textually, visually, graphically, etc.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students collect, organize, and analyze data from sources such as surveys and articles using computational tools, and generate visual models to communicate findings through graphs, charts, and infographics.

History and Social Science

- Students collect, organize, and analyze historical data related to a selected event, and generate visual representations (e.g., line graphs, bar graphs, choropleth maps) to illustrate changes over time or across regions. Students can then compare the effectiveness of each visual format and explain how different models communicate historical insights.

Mathematics

- Students will collect, organize, and analyze data on a personally meaningful topic, and generate multiple visual representations using computational tools to compare effectiveness and explain the appropriate use of each graph type.

Science

- Students conduct a digital science simulation using tools such as PhET to collect, organize, and analyze experimental data. They will generate visual models (e.g., line graphs, scatter plots) and interpret how variables interact, using data-driven evidence to explain conclusions about the scientific phenomenon.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

6.DA.3 The student will make predictions and draw conclusions from data visualizations.

a. Visualize data using appropriate graphs, charts, and data visualization techniques to enhance understanding and communicate findings effectively.
b. Use computational tools to analyze patterns within data sets and identify trends.
c. Draw conclusions and make predictions based on the analysis and interpretation of the data visualization.
d. Utilize simulations and models to formulate, refine, and test hypotheses.

Understanding the Standard

[6.DA.3a] Data visualization uses graphical techniques such as charts, graphs, and maps to represent data attributes through visual encoding. These techniques display relationships among quantitative and categorical variables, helping reveal trends, patterns, and outliers that may not be apparent in raw data. Choosing the appropriate visualization such as a bar chart for comparing categories, a line chart for time-based trends, or a scatter plot for identifying correlations supports accurate interpretation of visual outputs. Effective visualizations also enable audience-specific communication, making data-driven insights more accessible across different contexts.

[6.DA.3d] Models and simulations represent systems that are too complex, too large, or not directly observable in real time. Computer-based models use algorithms, rules, or equations to replicate the behavior of real-world systems. Simulations run these models to test outcomes under changing conditions, often using large-scale datasets that exceed human capacity for manual analysis. Applications include climate modeling, population growth, disease spread, and physical system design across fields such as science, engineering, and economics.

- A Model is a simplified representation of a real-world system, phenomenon, or process.
  Models are used to test hypotheses by generating data that can be evaluated for accuracy. Results help determine whether the model reflects real-world behavior and guide necessary adjustments. Models must undergo validation and refinement to improve reliability.

- A Simulation is a dynamic, virtual execution of a model that shows how a physical or conceptual system behaves under defined conditions.
  Simulations use data-driven models to replicate real-world scenarios. Examples include vehicle performance in varying weather conditions, projected population growth, outcomes of rocket launches, or the effectiveness of a vaccine against disease spread.

[6.DA.3bc] Drawing conclusions and making accurate predictions from data visualizations requires careful interpretation of patterns, trends, and relationships shown in the visual format. Recommended steps to follow:

1. Analyze the Visualization: Carefully examine the visual elements, such as the type of chart, axes, and data points.
2. Identify Trends and Patterns: Look for trends, correlations, and anomalies within the data.

3. Interpret the Findings: Explain the meaning of the observed trends and patterns in relation to the underlying data.
4. Draw Conclusions: Summarize the key insights derived from the analysis and support them with evidence from the visualization.
5. Make Predictions: Use the insights gained to forecast future trends or outcomes, considering potential limitations and uncertainties.

Concepts and Connections

CONCEPTS

Data analysis and interpretation involve selecting appropriate graphs, charts, and visualization techniques to present information clearly and effectively. Computational tools assist in detecting patterns and trends within datasets, enabling interpretation for conclusions and predictions. Simulations and models support the testing, refinement, and validation of hypotheses, contributing to deeper understanding and informed decision-making.

CONNECTIONS

Within the grade level/course: At this grade level, students make predictions and draw conclusions from data visualizations.(6.DA.3)
Vertical Progression: In grade 5, students create multiple data representations to make predictions and conclusions. (5.DA.2) In Grade 7, utilize computational tools to visualize and evaluate data to draw conclusions and make predictions.. (7.DA.1).

ACROSS CONTENT AREAS

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Mathematics

- **6.PS.1** The student will apply the data cycle (formulate questions; collect or acquire data; organize and represent data; and analyze data and communicate results) with a focus on circle graphs.

- **6.PS.2** The student will represent the mean as a balance point and determine the effect on statistical measures when a data point is added, removed, or changed.

DIGITAL LEARNING INTEGRATION

- **6-8.CT** Students communicate clearly and express themselves creatively for a variety of purposes using appropriate technologies (including assistive technologies), styles, formats, and digital media appropriate to their goals.
  A. Students select and use appropriate technologies to create, share, and communicate their work effectively, considering the audience.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

Mathematics

- Students collect, organize, and analyze data showing the relationship between ingredient quantities and recipe servings. Using computational tools, they will generate visual models to identify numerical patterns and predict serving outcomes. Students will then draw conclusions about proportional relationships and scaling, demonstrating mathematical reasoning and communication.

Science

- Students gather and organize seasonal environmental data, including local temperatures and precipitation levels. Utilizing computational tools (e.g. spreadsheet), students construct line and bar graphs to visualize changes over time. Through critical analysis of these visualizations, students will interpret climate patterns and predict their effects on local ecosystems such as shifts in plant growth cycles or animal behavior. Students will then formulate evidence-based conclusions and communicate the ecological implications using academic language and reasoning.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. FOSTERING COMPUTATIONAL THINKING PRACTICES:

5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems
2. Plan and Design Artifacts
3. Create, Communicate and Document Solutions

6.DA.4 The student will identify ways people curate and provide training data.

a. Identify and list ways people provide data that is used as training data.
b. Discuss the role of human intervention in curating training data.
c. Identify and describe the effect training data has on the accuracy of artificial intelligence systems.

Understanding the Standard

[6.DA.4a] Training data refers to the dataset used to build and adjust machine learning models. It provides structured input that allows models to identify patterns and make predictions. Sources include:

- Public datasets from government agencies, research institutions, and open repositories
- User-generated content such as social media posts, reviews, or multimedia uploads
- Sensor data from smartphones, wearables, or Internet of Things (IoT) devices
- Synthetic data generated by algorithms when real-world data is insufficient or sensitive

Training data must be accurate, diverse, and properly labeled to avoid bias and improve model performance. Data often undergoes preprocessing to ensure consistency. Training data is distinct from testing and validation data, which are used to evaluate the model after learning is complete.

[6.DA.4b] Human involvement plays a critical role in preparing training data by ensuring accuracy, consistency, and contextual relevance**.** Machine learning models rely on data that reflects real-world scenarios, and humans provide the judgment needed to evaluate quality, resolve ambiguity, and maintain alignment with the intended task. Without human oversight, training data may include errors, biases, or gaps that reduce model performance and reliability. Key processes include:

- Clean: Detects and corrects errors or inconsistencies in the dataset. This includes removing duplicate records, handling missing values, and standardizing formats.
- Label: Assigns predefined categories or tags to data points. In supervised machine learning, labels provide the correct output the model should learn from. For example, images may be labeled as "cat," "dog," or "bird" in a classification task.
- Annotate: Adds detailed metadata or contextual information to data points. This may include outlining objects in images, transcribing spoken language, or marking relevant phrases in text. Annotation provides models with deeper context and supports more complex tasks such as object detection or natural language understanding.

These steps reduce bias, remove irrelevant content, and ensure the data is suitable for training machine learning models or conducting accurate analysis.

[6.DA.4c] The quality and quantity of training data impacts the accuracy and reliability of AI systems**.** A well-curated, diverse dataset supports model generalization and consistent performance across different inputs. A dataset that is biased, incomplete, or limited leads to inaccurate outputs and unreliable behavior. Sufficient data volume and coverage help reduce error and support broader application of the model.

Concepts and Connections

CONCEPTS

Training data is the information used to teach artificial intelligence systems how to perform specific tasks or make decisions. It is collected from sources such as sensors, digital transactions, online activity, surveys, and user-generated content. Human involvement is critical for cleaning, labeling, and validating the data to ensure accuracy and minimize bias. The performance and reliability of AI systems depend on the quality and representativeness of their training data, as flawed or incomplete data can produce inaccurate or biased results.

CONNECTIONS

Within this grade level: At this grade level, students identify ways people curate and provide training data. (6.DA.4)
Vertical progression: In Grade 5,students explain the significance of training data in machine learning (5.DA.3). In Grade 7, students explain the process and application of data's role in machine learning and artificial intelligence. (7.DA.2).

ACROSS CONTENT AREAS

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Mathematics

- **6.PS.1** The student will apply the data cycle (formulate questions; collect or acquire data; organize and represent data; and analyze data and communicate results) with a focus on circle graphs.
- **6.PS.2** The student will represent the mean as a balance point and determine the effect on statistical measures when a data point is added, removed, or changed.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students examine curated textual datasets. Using spreadsheets or other computational tools, students will organize sample data and explore how word patterns contribute to model learning and accuracy. They will expand technical vocabulary by applying domain-specific terms such as “token,” and “annotation,” in context, and compose evidence-based explanatory essays that present their findings using claims supported by data examples and clear reasoning.

History and Social Science

- Students examine historical datasets such as migration, economic growth, or population change and simulate curating this data for AI applications by selecting, labeling, and refining relevant data points. They will analyze how data structure, accuracy, and variety affect the predictive capabilities of machine learning models.

Science

- Students explore how data such as temperature, rainfall, and pollutant levels are collected, curated, and prepared for use in AI models that predict weather and climate shifts. Students will simulate the process of selecting and refining data, evaluating how factors like accuracy, completeness, and labeling affect AI predictions. While doing so, students will apply technical vocabulary such as “data integrity,” “sensor input,” and “model training,” and compose evidence-based explanatory essays using data examples to support their conclusions.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Computational Thinking Practices:

5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data Sets

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-world problems

Impacts of Computing (IC)

6.IC.1 The student will assess the impact of computing technologies on local society.

a. Explain how computing impacts innovation and describe the development of new computing technologies in communication, entertainment, and business.
b. Discuss how computing technologies have influenced various industries and sectors locally.
c. Research simple and complex problems that computing systems can be used to solve.
d. Analyze the implications of emerging technologies and potential real-world impact in the local community.

Understanding the Standard

Computing technology has played a central role in shaping and responding to societal change, particularly since the mid-20th century. Building on centuries of technological advancement, the development of computing systems introduced new capabilities in data processing, automation, and communication. As computing power has exponentially increased from mainframes to cloud computing and embedded systems, its influence has expanded across all sectors, enabling advancements in fields such as artificial intelligence, cybersecurity, digital communication, and data analytics. Today, computing technology underpins critical infrastructure, drives economic transformation, and shapes how individuals interact, learn, and work in a globally connected society.

[6.IC.1a] Computing technology has transformed communication by enabling instantaneous, global information exchange through email, messaging platforms, video conferencing, and social media. These tools have redefined interpersonal interaction, removed geographic barriers, and facilitated real-time collaboration across industries. Data compression, network protocols, and cloud-based services support scalable communication systems, allowing individuals and organizations to transmit text, audio, and video at unprecedented speed and reliability.

In entertainment, computing has reshaped content creation, distribution, and user engagement. Digital animation, game engines, and generative AI expand creative possibilities in film, gaming, and music production. Streaming platforms rely on advanced algorithms to deliver personalized content through predictive modeling and user data analysis. Interactive experiences are driven by high-performance computing and graphics processing, merging physical and digital environments for immersive media consumption.

In business, computing technologies optimize operations, decision-making, and customer engagement. Enterprise systems use real-time data analytics to monitor performance, manage supply chains, and forecast trends. Artificial intelligence automates tasks such as fraud detection, inventory management, and customer support. Cloud computing enables scalable infrastructure, reducing physical hardware needs while supporting remote work and global access. These capabilities increase productivity, reduce costs, and support innovation in competitive markets.

[6.IC.1b] Computing technology has transformed nearly every industry, including manufacturing, retail, finance, healthcare, and transportation. The integration of computing systems has redefined operational processes, enabled automation, and increased precision, scalability, and decision-making capabilities. Core technologies such as sensors, GPS, cloud computing, databases, and the Internet serve as foundational infrastructure, supporting real-time data collection, storage, and transmission. Advances in cybersecurity and digital forensics ensure data integrity and protect digital assets in an increasingly interconnected world. Emerging innovations to introduce new paradigms for how goods are produced, services are delivered, and users interact with digital environments. These technologies rely on the convergence of hardware, software, and data to enable adaptive, intelligent systems that respond dynamically to changing conditions and user needs.

[6.IC.1 c] Computing has also impacted innovation. Examples of innovation that have affected different career fields include providing the opportunity for faster production, accessing large quantities of data and information, advancing our ability in communication and collaboration, and self-publishing.

[6.IC.1 d] Computing systems are essential tools for solving both simple and complex problems across various domains. Simple tasks such as performing basic mathematical calculations, scheduling automated emails, and sorting data are efficiently handled by algorithms and software applications. More complex problems require advanced computational techniques, including the analysis of large datasets, pattern recognition, and predictive modeling. For example, computing systems can process historical data to forecast future events, generate simulations to model climate change, and extrapolate current trends to support long-term planning and strategic decision-making. These capabilities enable individuals and organizations to approach problem-solving with increased speed, accuracy, and scalability.

- Communication industry technological advancements include but are not limited to: social media, machine learning, Internet of things, driverless cars, security and privacy, networking, branding, funnel marketing.
- Technological advancements in Business industry may include but are not limited to: traceability and safety software, database advancements, customer scheduling, big data and machine learning, automation, and augmented reality.
- Entertainment industry technological advancements may include but are not limited to: downloading digital music, video streaming, ticket sales, marketing and receiving, licensing, voice technology, and holograms.

The interaction between technology and society has continuously influenced historical, economic, and cultural development. As technological advancement accelerates, its effects on industries, systems, and daily life expand in scope and complexity. Understanding the influence of technology enables individuals and institutions to apply its capabilities strategically while identifying and addressing associated risks.

Concepts and Connections

CONCEPTS

Computing drives innovation by enabling the creation of new technologies across sectors such as communication, entertainment, and business. These advancements reshape local industries by transforming processes, services, and user interactions. Computing systems support problem-

solving at all levels from basic tasks like data entry to complex functions such as predictive modeling and automation. Analyzing emerging technologies is essential for understanding their impact on communities and broader societal systems.

CONNECTIONS

Within this grade level: At this grade level, students assess the impact of computing technologies on local society. (6.IC.1).
Vertical progression: In Grade 5,students examine the effects of social interactions due to computing technologies. (5.IC.5). In Grade 7, students expand their assessment of the impact of computing technologies beyond local society to consider the global impact as well. (7.IC.1).

ACROSS CONTENT AREAS

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Science

- **6.9** The student will investigate and understand that humans impact the environment, and individuals can influence public policy decisions related to energy and the environment. Key ideas include natural resources are important to protect and maintain; renewable and nonrenewable resources can be managed; major health and safety issues are associated with air and water quality; major health and safety issues are related to different forms of energy; preventive measures can protect land-use and reduce environmental hazards; and there are cost/benefit tradeoffs in conservation policies.

DIGITAL LEARNING INTEGRATION

- **6-8.GC** Students use appropriate technologies, including assistive technologies, to broaden their perspectives and enrich their learning by collaborating with others and working effectively in teams locally and globally.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students analyze the impact of computing technologies on cultural norms and communication within their local community. Students investigate digital trends and curate datasets to represent patterns in media use, assessing how platform design, content algorithms, and interaction formats influence public dialogue. Applying terms like "data aggregation," "digital influence," and "user behavior," students will construct explanatory essays that present their findings.

History and Social Science

- Students research and document how specific computing devices such as early mobile phones, legacy gaming consoles, or foundational software impacted individual experiences and broader community practices. This investigation engages students in historical inquiry and digital literacy as they examine changes in functionality, accessibility, and cultural significance over time.

Science

- Students investigate and model how local governments and businesses apply computing technologies such as sensors, satellite imagery, and AI models to monitor factors within their community. Students analyze the functionality and development of tools like pollution tracking systems and smart waste management, exploring how algorithms and data processing enable environmental problem-solving.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
2. Safeguard Well-Being of Self and Others
3. Evaluate Resources and Recognize Contributions

6.IC.2 The student will analyze the impact of screen time on physical and mental health.

a. Analyze and describe the impact of excessive technology usage may have on one's physical health.
b. Examine the impact of blue light on sleep patterns and regulations.
c. Propose strategies that provide alternatives of technology usage to promote physical activity.
d. Discuss the potential impact the use of social media may have on self-identity and mental health.
e. Define cyberbullying and its impact on one's health and well-being.
f. Discuss the possible effects of cyberbullying.
g. Identify ways to report illegal or psychologically maladaptive online behavior.

Understanding the Standard

Screen time refers to the amount of time spent interacting with devices that have digital screens, such as smartphones, tablets, computers, and TVs. This includes both passive activities (e.g., watching videos) and interactive ones (e.g., using social media or gaming). According to the National Institutes of Health (NIH), high screen time can begin early and may influence social development from a young age.
Research shows that while screen time provides new ways to connect, excessive or unbalanced use can reduce quality time spent with family and friends, impacting social development (American College of Sports Medicine, 2017).

[6.IC.2abc] Excessive screen time can significantly impact both physical and mental health.

- Prolonged sitting, often associated with extended technology use, can lead to various physical health issues, such as poor posture, eye strain, and musculoskeletal disorders.
- The blue light emitted by electronic devices can disrupt sleep patterns, leading to insomnia and other sleep-related problems.

[6.IC.2d] Social media platforms, while offering numerous benefits, can also negatively impact mental health.

- Excessive use of social media can contribute to feelings of loneliness, depression, and anxiety
- Constant exposure to curated images and idealized lifestyles can lead to body image issues and low self-esteem.
- Cyberbullying, a form of harassment that occurs online, can have severe psychological consequences, including anxiety, depression, and even suicidal thoughts.

To mitigate the negative impacts of technology, it's essential to adopt healthy habits.

- Regular physical activity, such as exercise and outdoor recreation, can help counteract the sedentary nature of screen time.
- Limited screen time, especially before bed, can improve sleep quality.
- Practicing mindfulness techniques, such as meditation and yoga, can help reduce stress and anxiety.

[6.IC.2efg] Responsible technology use requires awareness of both content and time spent on digital platforms. Monitoring screen time and evaluating the quality and source of online content supports healthier engagement. In cases of cyberbullying, incidents should be reported immediately to a trusted adult, relevant authorities, or platform administrators. Proactive management of digital behavior helps reduce potential harm and supports safe, constructive participation in online environments.

Teachers should review school division's Acceptable Use Policy (AUP), student handbook, and code of conduct to identify policies related to cyberbullying, online harassment, and responsible technology use. These documents can serve as the foundation for designing lessons that address digital citizenship, legal implications, and personal safety. Incorporating real-world scenarios and aligning instruction with local policies ensures students understand both their rights and responsibilities in digital spaces.

Concepts and Connections

Concepts

Screen time can affect physical and mental health through issues like screen fatigue, disrupted sleep, and harmful online interactions such as cyberbullying. Promoting healthy habits, recognizing the effects of misuse, and knowing how to report harmful behavior are essential for maintaining well-being.

Connections

Within this grade level: At this grade, students analyze the impact of screen time on physical and mental health. (6.IC.2)
Vertical progression: In Grade 5, students examined the impact of screen time on academic performance. They will compare and contrast screen time that benefits and hinders academic performance (5.IC.2). In Grade 7, students examine the impact of screen time on their interactions with others and create a social media plan demonstrating safe and responsible use. (7.IC.2)

Across Content Areas

History and Social Science

- CE.1 The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

DIGITAL LEARNING INTEGRATION

- **6-8.KC** Students critically curate a variety of digital resources using appropriate technologies, including assistive technologies, to construct knowledge, produce creative digital works, and make meaningful learning experiences for themselves and others.
  A. Students practice and demonstrate the ability to effectively use research strategies to locate appropriate primary and secondary digital sources in a variety of formats to support their academic and personal learning and create a research product.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students read and evaluate a selection of texts articles, essays, and narratives focused on the impact of social media. Students work collaboratively to translate their insights into action by designing a digital well-being campaign or resource kit (e.g., infographics, short videos, or app mockups) that promotes healthy social media habits for peers, demonstrating both critical thinking and creative application of computing principles.

History and Social Science

- Students investigate the issue of cyberbullying by defining key terms and examining common forms such as harassment, exclusion, and impersonation along with the psychological impact on individuals, including anxiety, depression, and social isolation. They will engage in critical discussion on strategies for reporting cyberbullying, including digital reporting tools and seeking support from trusted adults.

Science

- Students investigate the science behind blue light and its physiological effects, focusing on how screen exposure disrupts sleep patterns and circadian rhythms. They will analyze research on light wavelengths and digital device usage, then evaluate strategies for minimizing blue light impact

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. FOSTERING COMPUTATIONAL THINKING PRACTICES:

1. Decompose Relevant Problems

C. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems
2. Plan and Design Artifacts
3. Create, Communicate, and Document Solutions

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
2. Safeguard Well-Being of Self and Others

6.IC.3 The student will explore career pathways and identify how computer science and computational thinking practices align with these pathways.

a. Investigate a career of interest and determine how computer science and computational thinking practices are used in the chosen career.

Understanding the Standard

[6.IC.3 a] Numerous careers rely on data collection, analysis, and interpretation. Roles such as data analyst, data scientist, data engineer, and data architect are in high demand across sectors including finance, healthcare, logistics, marketing, and public policy. The application of computer science knowledge and computational thinking extends beyond traditional technology careers. Fields such as agriculture use automation and sensor data to optimize crop production, while education relies on learning analytics to inform instruction. In healthcare, algorithms support diagnostics and patient care, and in the arts, creative coding enables interactive digital media.

Core computer science skills: such as abstraction, algorithmic design, iterative testing, and digital security are foundational to problem-solving in these domains. Real-time data processing, modeling, and system optimization require professionals who can think computationally, interpret data critically, and apply solutions that are efficient, ethical, and scalable. The integration of computer science with domain-specific expertise is essential for addressing complex problems in a data-driven global economy.

Examples of these careers include business or financial analysts who use data to improve investments and outcomes for individuals or businesses, medical personnel who must manage and secure healthcare information systems, and researchers who maintain databases and interpret results to identify patterns and potential solutions.

It is important to explore various aspects of potential careers such as work expectations, pay rate, and required education as they explore potential career paths Current information on education, pay, and employment projections can be found through the U.S. Bureau of Labor Statistics (https://www.bls.gov/emp/).

Concepts and Connections

CONCEPTS

Computer science concepts and skills apply across a wide range of careers in various industries. Exploring different professions helps students identify how these skills align with job responsibilities and how computational thinking contributes to problem-solving, efficiency, and innovation.

CONNECTIONS

Within the grade level/course: At this grade level, students explore career pathways and identify how computer science and computational thinking practices align with these pathways. (6.IC.3)
Vertical Progression: In Grade 5, students identify the impact of computing technologies on the workforce, culture, and global society. (5.IC.3). In Grade 7, students self-assess and examine individual preferences and relate them to a chosen computer science career. (7.IC.3).

ACROSS CONTENT AREAS

English

- **6.DSR** The student will build knowledge and comprehension skills from reading a range of challenging, content-rich texts. This includes fluently reading and gathering evidence from grade-level complex texts, reading widely on topics to gain purposeful knowledge and vocabulary, and using reading strategies when comprehension breaks down.
- **6.RI** The student will use textual evidence to demonstrate comprehension and build knowledge from grade-level complex informational texts read.

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

DIGITAL LEARNING INTEGRATION

- **6-8.KC** Students critically curate a variety of digital resources using appropriate technologies, including assistive technologies, to construct knowledge, produce creative digital works, and make meaningful learning experiences for themselves and others.
  A. Students practice and demonstrate the ability to effectively use research strategies to locate appropriate primary and secondary digital sources in a variety of formats to support their academic and personal learning and create a research product.

Opportunities for Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

History and Social Science

- Students will explore a career of personal interest such as healthcare, engineering, graphic design, or business and examine how computer science and computational thinking are integrated into the profession. They will research specific applications of digital technologies, including tools like electronic medical records, CAD software, data analytics, or algorithmic marketing, and analyze how these influence daily workflows and problem-solving. Applying technical vocabulary like "automation," "data visualization," or "computational modeling," students will design a digital presentation or visual report highlighting key CS-related skills, workplace technologies, and their societal impact.

Mathematics

- Students will explore careers that involve data science, analytics, or other math-intensive computing fields such as sports statistics, finance, cybersecurity, machine learning, and market research. Through this investigation, they will identify how mathematical skills (e.g., statistics, algebra, probability) and computational thinking strategies (e.g., pattern recognition, data abstraction, algorithmic reasoning) are used to process information, uncover insights, and drive decision-making.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

3. Evaluate Resources and Recognize Contributions

6.IC.4 The student will identify copyrighted and licensed software material.

a. Identify the role of software licenses, including open-source, and why they are used.
b. Compare and contrast the positives and negatives of various software licenses.

Understanding the Standard

[6.IC.4a] Software licenses are legal agreements that define how software can be used, distributed, and modified.

[6.IC.4 b] These licenses protect the intellectual property rights of developers while outlining the permissions and limitations for users.

Common types of software licenses include, but are not limited to:

- Open-source software licenses grant users the freedom to use, modify, and distribute the software. This fosters collaboration, innovation, and the creation of high-quality software. However, open-source software may require technical expertise to install and configure, and it may not always offer the same level of commercial support as proprietary software.

- Proprietary software licenses restrict the use and distribution of software to specific terms and conditions. While proprietary software often offers excellent technical support and regular updates, it can be more expensive and may not allow for customization or modification. Understanding the differences between open-source and proprietary software licenses is crucial for making informed decisions about software usage and development.

- Shareware software licenses provide a trial, typically for 30 days, that allows users access to software. After the trial period users must purchase a license to continue using the program. This allows for users to test software before committing to purchasing the full software license.

Concepts and Connections

CONCEPTS

Software licenses specify how programs may be used, distributed, and modified, including models such as open-source and proprietary. These licenses protect intellectual property and establish the legal boundaries for software use and sharing.

CONNECTIONS

Within this grade level: At this grade level, students identify software license types: open-source, proprietary, and shareware and how each affects software use, distribution, and modification. and users. (6.IC.4)

Vertical progression: In Grade 5, students will learn the difference between open-source licenses and copyright. (5.IC.4). In Grade 7,students discuss and describe intellectual property protections and list safeguards used to prevent intellectual property infringement. (7.IC.4).

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

History and Social Science

- Students research and examine the evolution of intellectual property and its impact on digital innovation through a historical and social science perspective. They will explore how legal and societal attitudes toward software ownership have developed since the rise of the digital age, tracing key moments such as the emergence of copyright law, the birth of open-source culture, and the commercialization of proprietary software.

Science

- Students explore the role of open-source software in advancing scientific research and experimentation. They will investigate how open-source tools are used in disciplines such as biology, physics, computer science, and astronomy to support simulations, data analysis, modeling, and collaborative problem-solving.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
3. Evaluate Resources and Recognize Contributions

6.IC.5 The student will describe the impacts of computing network architecture, including the role of the Internet in society.

a. Discuss ethical issues and laws related to accessibility, censorship, privacy, access, and safety while using the Internet.
b. Explain the role broadband connectivity has in social life, culture, and global economy.

Understanding the Standard

The Internet has revolutionized communication, work, and learning by enabling real-time access to information and global connectivity. It consists of interconnected servers, routers, and transmission media that facilitate the efficient exchange of data across networks. Beyond its technical structure, the Internet plays a central role in modern society by shaping how individuals interact, access services, and participate in civic life. This infrastructure supports advancements across sectors, including telemedicine in healthcare, virtual learning environments in education, digital finance, and cloud-based collaboration in business and research.

[6.IC.5 a] With the benefits of the internet come ethical concerns. Issues such as privacy, security, and digital divide have become increasingly important. It is crucial to be aware of these issues and take steps to protect oneself online [Refer to Computer Science 6.CYB.1 and 6.CYB.2].

Laws and regulations are being enacted to address these concerns, but technological advancements often outpace legal frameworks. Examples of such laws include:

- The Children's Online Privacy Protection Act (COPPA) - a federal law that gives parents control over what data is collected from their children under the age of 13. COPPA requires online entities to obtain parental consent before collecting, using or sharing a child's personal information. COPPA applies to websites, mobile apps and other physical items embedded with the ability to interact with apps or the internet such as smart watches, smart speakers and other smart appliances.
- Americans with Disability Act (ADA) - governs how adaptations and accommodations are made for people with disabilities. It prohibits discrimination against people with disabilities. In terms of computing, websites and apps must be equally accessible to both people with disabilities as they are for those without. Broadband connectivity, a high-speed internet connection, plays a vital role in modern society. It provides individuals with access to information, ability to connect with others, and participate in the global economy.

(6.IC.5 b) Broadband access has enabled remote work, online education, virtual learning, and e-commerce, transforming how individuals live, work, and engage with services. Limited access to high-speed internet contributes to the digital divide, where individuals or communities experience limited opportunities due to insufficient connectivity. Barriers such as geographic isolation, high service costs, and underinvestment in broadband infrastructure restrict access to reliable, high-quality internet, reinforcing existing social and economic disparities.

Concepts and Connections

CONCEPTS

Internet use involves key ethical and legal considerations such as accessibility, censorship, privacy, safety and access. These factors shape technology governance and user behavior, while broadband connectivity influences global social interaction, cultural exchange, and economic development.

CONNECTIONS

Within this grade level: At this grade level, students describe the impacts of computing network architecture, including the role of the Internet in society. (6.IC.5)

Vertical progression: In Grade 5, students examine the advances in computing technologies and their impact on communication and collaboration on a personal level and their impact on social interactions.(5.IC.5).

ACROSS CONTENT AREAS

English

- **6.LU** The student will use the conventions of Standard English when speaking and writing differentiating between contexts that call for formal English and situations where informal discourse is more appropriate.

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.
- **CE.12** The student will apply social science skills to understand the United States economy by a) describing the characteristics of the United States economy, including limited government, private property, profit, markets, consumer sovereignty, and competition; b) describing how in a market economy supply and demand determine prices; c) describing the types of business organizations and the role of entrepreneurship; d) explaining the circular flow that shows how consumers (households), businesses (producers), and markets

interact; e) explaining how financial institutions channel funds from savers to borrowers; and f) analyzing the relationship of Virginia and the United States to the global economy, with emphasis on the impact of technological innovations.

Digital Learning Integration

- **6-8.KC** Students critically curate a variety of digital resources using appropriate technologies, including assistive technologies, to construct knowledge, produce creative digital works, and make meaningful learning experiences for themselves and others.
  A. Students practice and demonstrate the ability to effectively use research strategies to locate appropriate primary and secondary digital sources in a variety of formats to support their academic and personal learning and create a research product.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students research the impact of the Internet on education, focusing on the advantages and challenges of online learning. They can investigate online educational platforms, e-books, and digital classrooms, and then create a presentation or report on how Internet access has improved learning opportunities. They can also explore issues related to digital equity, such as the impact of not having broadband access on students' ability to learn. The report could include suggestions on how broadband access could be improved to support education in underserved communities.

History and Social Science

- Students investigate the ethical issues surrounding the Internet, including privacy concerns, censorship, and online safety. They can research laws and policies that protect Internet users, such as the Children's Online Privacy Protection Act (COPPA) or the General Data Protection Regulation (GDPR). Students will then write a report or give a presentation on how ethical issues and laws impact users' ability to access the Internet safely and how these laws help protect people from online risks.

Mathematics

- Students collect data on how broadband Internet access impacts businesses in different regions, particularly in rural versus urban areas. They can explore how businesses that rely on high-speed Internet for e-commerce, cloud services, or video conferencing benefit from broadband connectivity. After collecting data, students will create graphs and charts to compare the economic outcomes of businesses with access to broadband versus those without. Students will analyze the results to understand how broadband connectivity can support or hinder economic development.

Science

- Students research how devices like smart thermostats, wearable health trackers, and smart home appliances use the Internet to improve everyday life. They will explore how broadband connectivity allows these devices to communicate and work together in a network, improving convenience and efficiency. Students will create a diagram showing how data flows through different IoT devices and explain how this connectivity enhances both individual lives and society as a whole.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
2. Safeguard Well-Being of Self and Others
3. Evaluate Resources and Recognize Contributions

6.IC.6 The student will investigate and analyze the impact of the progression and advancement of AI technologies on industries.

a. Discuss the type of industries that may be impacted by the use and integration of Artificial Intelligence (AI).
b. Compare and contrast the evolving nature of work across diverse industries because of the progression and advancement of Artificial Intelligence.

Understanding the Standard

Artificial intelligence (AI) is a branch of computer science that aims to create intelligent agents, which are systems that can reason, learn, and act autonomously. An intelligent agent is software that has been designed to use artificial intelligence and machine learning to perform tasks without human intervention. AI systems are designed to mimic human intelligence, enabling them to perform tasks that would typically require human cognitive abilities.

[6.IC.6a] Artificial intelligence (AI) is transforming industries worldwide by enabling machines to perform tasks that traditionally require human intelligence. In sectors such as healthcare, finance, transportation, and manufacturing, AI is being integrated to enhance decision-making, automate complex processes, and increase efficiency. Its widespread adoption is reshaping how systems operate and redefining the nature of work and daily life.

- In healthcare, AI-powered tools are being used to analyze medical images, detect diseases early, and develop personalized treatment plans.
- In the financial industry, AI algorithms are employed to detect fraud, assess creditworthiness, and automate trading.
- In the automotive industry Ai is being used to develop self-driving cars and improve vehicle safety.
- In education, AI have the ability to provide intelligent tutoring systems. These systems use AI algorithms to adapt to each student's individual learning pace and style, providing personalized instruction and feedback. This allows students to learn at their own pace and receive targeted support where needed.

[6.IC.6b] The integration of artificial intelligence into the workplace can increase efficiency, streamline operations, and drive innovation across industries. By automating repetitive and routine tasks, AI allows professionals to focus on higher-level skills such as creativity, critical thinking, and complex problem-solving. Its widespread adoption also presents potential challenges. Job displacement may occur in roles heavily dependent on manual or administrative tasks. Ethical concerns including algorithmic bias, lack of transparency in decision-making, and data privacy require oversight. The evolving nature of work demands continuous upskilling and adaptability

Concepts and Connections

CONCEPTS

Artificial intelligence (AI) impacts diverse industries such as healthcare, finance, transportation, and manufacturing by changing how tasks are executed and decisions are made. As AI technology advances, the evolving nature of work demands analysis of both the emerging opportunities and the challenges it presents across sectors.

CONNECTIONS

Within this grade level: At this grade level, students investigate and analyze the impact of the progression and advancement of AI technologies on industries. (6.IC.6)
Vertical progression: In Grade 7, students evaluate the impact of Artificial Intelligence in various professions.(7.IC.5).

ACROSS CONTENT AREAS

English

- **6.LU** The student will use the conventions of Standard English when speaking and writing differentiating between contexts that call for formal English and situations where informal discourse is more appropriate.

History and Social Science

- **CE.1**.The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students examine how English language is predominantly used in the development and training of AI systems, including chatbots, virtual assistants, translation engines, and recommendation algorithms. They'll explore how linguistic choices and communication styles embedded in training data affect how AI understands, responds to, and represents human language.

History and Social Science

- Students investigate how artificial intelligence is transforming major industries such as healthcare, transportation, and manufacturing through the lens of historical change and social impact. Students examine how traditional job roles have evolved over time in response to technological advancements, comparing legacy tasks with emerging roles influenced by AI tools like diagnostic algorithms, autonomous vehicles, and intelligent robotics.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

A. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-World Problems

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices
2. Safeguard Well-Being of Self and Others
3. Evaluate Resources and Recognize Contributions

Networks and the Internet (NI)

6.NI.1 The student will outline the advantages and disadvantages of transmitting information over the Internet, including speed, reliability, cost, and security.

a. Explain the role of the Internet in social life, culture, and the economy.
b. Explain data transfer and the impact of connectivity speed when data is going from one device to another.
c. Compare the speed and reliability of various data transmission media.
d. Describe the advantages and disadvantages of transporting information over the Internet.

Understanding the Standard

[6.NI.1a] The internet has revolutionized social life, culture, and the economy. It connects people across the globe, fostering communication and collaboration more quickly than historically possible. It has transformed how we consume information, entertainment, and news. Additionally, it has driven economic growth, enabling e-commerce, remote work, and digital innovation.

[6.NI.1b] The speed and reliability of data transfer are critical to the performance and efficiency of internet-based services. Data transfer refers to the movement of digital information between devices or systems across a network. Transmission speed, typically measured in megabits per second (Mbps), determines how quickly data can be sent or received. Higher speeds support smoother streaming, faster web browsing, shorter download times, and reduced latency during video conferencing or online gaming. Connectivity speed directly impacts user experience and system responsiveness, especially in data-intensive applications. Unlike physical delivery of documents or media, the Internet enables rapid digital transfer at scale and low cost, making real-time communication and cloud-based services possible. Limitations in speed or reliability such as bandwidth congestion or poor infrastructure can hinder productivity and access to critical online resources.

- There are a variety of factors to consider when evaluating transmission speed, including the size of the file or artifact, available bandwidth (maximum rate of data transfer), and concurrent uploading or downloading activity taking place on a given network.

[6.NI.1c] Several factors influence the speed and reliability of data transmission over the Internet, including file type and size. Larger files generally require more bandwidth and time to transfer, especially over slower or unstable connections. Different file types vary in size and complexity, which affects how efficiently they can be transmitted. For example, a text-based file such as a .doc (Microsoft Word) or .pdf (Portable Document Format) typically transfers faster than multimedia files like .mp3 (audio), .avi (video), or .gif (animated image). Compressed formats such as .zip (ZIP Archive) reduce file size for more efficient transfer, while .html files are optimized for quick loading in web environments. Comparing these formats helps illustrate how data type, compression, and content influence overall transmission performance.

[6.NI.1d] While the Internet provides numerous advantages, there are also potential risks and limitations associated with transmitting data or files across it:

- Security Risks:
  - Data Breaches: Sensitive information can be intercepted and stolen.
  - Cyberattacks: Systems can be compromised, leading to data loss or disruption of services.
- Privacy Concerns:
  - Data Tracking: Online activities can be monitored and analyzed to build user profiles.
  - Identity Theft: Personal information can be misused to steal identities.
- Reliability Issues:
  - Network Outages: Internet connectivity can be disrupted, hindering data transmission.
  - Slow Speeds: Slower speeds can impact productivity and online experiences.

Individuals should implement strong security protocols, monitor online activity carefully, and remain informed about emerging threats and current cybersecurity best practices to reduce these risks and concerns.

Concepts and Connections

CONCEPTS

The Internet enables communication, cultural exchange, and economic activity through the transfer of data between connected devices. Transmission speed and media type directly affect the efficiency and reliability of data transfer. Analyzing the advantages and limitations of online data transmission helps assess performance, security, and accessibility implications.

CONNECTIONS

Within the grade level/course: At this grade level, students learn how the internet has revolutionized communication, culture, and the economy, and the importance of data transmission speed for efficient online activities. (6.NI.1)
Vertical Progression: In Grade 5, students identify and describe cloud computing. (5.NI.1) In Grade 7, students describe why protocols are needed for data transmission and further expand the process for sending files over the internet. (7.NI.1).

ACROSS CONTENT AREAS

History and Social Science

- **CE.1** The student will demonstrate skills for historical thinking, geographical analysis, economic decision making, and responsible citizenship by a) analyzing and interpreting evidence from primary and secondary sources, including charts, graphs, and political cartoons; b) analyzing how political and economic trends influence public policy, using demographic information and other data sources; c) analyzing information to create diagrams, tables, charts, graphs, and spreadsheets; d) determining the accuracy and validity of information by separating fact and opinion and recognizing bias; e) constructing informed, evidence-based arguments from multiple sources; f) determining multiple cause-and-effect relationships that impact political and economic events; g) taking informed action to address school, community, local, state, national, and global issues; h) using a decision-making model to analyze and explain the costs and benefits of a specific choice; i) applying civic virtue and democratic principles to make collaborative decisions; and j) defending conclusions orally and in writing to a wide range of audiences, using evidence from sources.

Opportunities for Computer Science Integration

Curriculum integration strengthens conceptual understanding and skill application. This can be done through multidisciplinary, interdisciplinary, and transdisciplinary approaches to integration. The examples below illustrate multiple ways to integrate computer science.

English

- Students write an expository essay analyzing how Internet speed affects everyday digital activities—including homework completion, social media usage, online gaming, and video streaming. They will explain how slower connectivity can disrupt or delay these tasks, while faster speeds enhance overall user experience by enabling quick access, smooth interactions, and uninterrupted content delivery.

Mathematics

- Students apply mathematical concepts such as rates, unit conversions, and data interpretation to analyze Internet transmission media including fiber-optic, 4G, 5G, and Wi-Fi. Students collect real-world data on download/upload speeds and cost per gigabyte, organizing the information into tables and calculating performance metrics using formulas like Cost per MB = Total cost ÷ Total data transferred. Students will then create bar or line graphs to compare transmission media and identify trends in speed and cost efficiency.

Skills in Practice

Students should engage in the following practices to deepen their conceptual understanding and enhance the application of skills aligned with the Computer Science *Standards of Learning*. These practices are explained in more detail in Appendix A.

B. Fostering Iterative Design Practices:

1. Identify, Define, and Evaluate Real-world Problems

D. Fostering Digital Literacy Practices:

1. Responsible Use Practices

Appendix A

Grade 6-12 Computer Science Skills and Practices Continuum

Students develop essential practices: collaboration, computational thinking, iterative design, and digital literacy. Students use these practices to engage with core computer science concepts, create artifacts, and problem-solve across disciplines. Artifacts can include but are not limited to prototypes, programs, planning documents, animations, or abstractions. Abstractions can include but are not limited to visualizations, storyboards, flowcharts, infographics, generalizations, decision trees, models, or computer simulations.

A. **Fostering Collaborative Computing Practices**
1. **Cultivate Relationships and Norms:** • **All levels**: Students take turns in different group roles. Recognize group member strengths, ask clarifying questions, and practice self-advocacy. • **Skill progression:** Students also rotate roles and encourage participation. Evaluate group dynamics and improve teamwork. Advocate for the needs of others, such as users needing assistive technology.
2. **Include Multiple Perspectives:** • **All levels:** Students discuss design choices, compare preferences and choices, and incorporate ideas from peers into designs. Ask questions, and seek input from users with diverse abilities, experiences, and perspectives. Reflect on how perspectives aid problem-solving across contexts. • **Skill progression:** Students also integrate user-centered considerations and test with varied audiences throughout the design process. Apply systems thinking, questioning, and data to improve designs.
3. **Create and Accept Feedback:** • **All levels:** Students seek and provide constructive feedback by asking peers probing questions like "why do you think that?" and sharing ideas to help improve. Practice active listening and paraphrasing. Begin to use evidence to support feedback and distinguish opinions. Reflect on how feedback aids problem-solving across contexts. • **Skill progression:** Students also tailor constructive feedback to support peers' stated goals. Create action steps from feedback to optimize designs and improve problem-solving practices, such as collaboration.
4. **Select and Use Collaboration Tools:** • **All levels:** Students use collaboration tools like whiteboards, digital tools, and paper. Use computational thinking practices like sequencing (algorithms) and representations (abstractions) to collaboratively plan and create. Use timelines and begin to make decisions about which collaboration tools to use. Reflect on what strategies worked as predicted and ways to improve.

- **Skill progression:** Students also choose collaboration tools and evaluate tradeoffs of various methods of project management. Refine through repeated cycles of development and feedback.

Instructional Considerations for Collaboration Practices

Possible **instructional approaches** to foster collaboration practices:

1. Design instruction around authentic problems that require collaboration. Assign roles, provide clarifying and probing question stems, and model strategies students can use to identify and advocate for their needs.
2. Provide resources to support exploring diverse viewpoints and end users. Model curiosity, perspective-taking, and empathy.
3. Model sentence stems for constructive feedback, establish routines for self and group reflection, and practice incorporating diverse viewpoints. Implement pair programming with opportunities to practice giving and receiving feedback.
4. Model tool selection and project management structures. Provide opportunities to practice various methods and reflect.

Instructional activities may include but are not limited to:

- **Classroom Discussions:** Organize discussions that engage students in respectful discourse to acknowledge opposing perspectives.
- **Timeline Creation:** Have students create or evaluate and modify timelines that illustrate the steps needed to complete a task as a group.
- **Simulated Shark Tank Innovation Challenge:** Create an innovation design challenge where students collaboratively apply computer science content to solve a problem or launch a new idea.
- **Case Studies:** Provide case studies of design decisions that real computer scientists face and have students analyze and present their recommended choices based on computer science content knowledge.

B. Fostering Computational Thinking Practices

1. **Decompose Relevant Problems:**

- **All levels:** Students break problems, information, and processes into parts. Identify relationships and connections among parts. Reflect on how decomposition aids problem-solving across contexts. Begin to identify subproblems. Apply systems thinking to explore interdisciplinary connections,
- **Skill progression:** Students further break problems into subproblems and integrate existing solutions or procedures. Apply algorithms, such as searching and sorting, to recursively break a problem into similar subtasks that can be solved and combined to solve the main problem.

2. **Explore Common Features and Relationships to Extract Patterns:**

- **All levels:** Students recognize, organize, and describe patterns in data; include relationships and repeated sequences (loops). Explore how visualizations can show relationships and patterns in data. Reflect on how pattern analysis aids problem-solving across contexts. Analyze patterns to develop generalizations and models and test their limits. Use patterns to identify trends and make predictions.

- **Skill progression:** Students also apply pattern analysis to validate inputs, analyze trends, justify design decisions, and create artifacts.

3. **Apply Abstraction to Simplify, Represent, and Problem Solve:**

- **All levels:** Students use and/or create abstractions (e.g. visualizations, storyboards, flowcharts, generalizations, decision trees, models, computer simulations) to simplify problems, represent information, organize thinking, communicate, and create artifacts. Identify and ask questions about the information hidden in an abstraction. Reflect and compare abstractions to explore strengths, limitations, and how they impact problem-solving across contexts.
- **Skill progression:** Students intentionally use abstractions to support throughout the problem-solving process to aid in understanding, planning, and predictions. Incorporate modularity into programs. Create models of complex systems to inform design solutions and understand how and why parts connect. Evaluate and systematically test the limits, opportunities, and uses of models and abstractions.

4. **Apply Algorithmic Thinking to Problem Solve and Create:**

- **All levels:** Students use algorithmic thinking to develop a sequence of steps to plan, create, test, and refine computational artifacts with and without technology. Reflect and identify ways algorithmic thinking aids problem-solving across contexts.
- **Skill progression:** Students use pseudocode and generalizations to organize, create and seek and incorporate feedback on more complex designs. Include automated solutions. Analyze algorithmic solutions and systematically test their limits. Validate their outputs and optimize for design criteria like accessibility.

5. **Apply Computational Thinking Practices to Select, Organize, and Interpret Data:**

- **All levels:** Students use patterns and algorithmic thinking to organize data, identify trends, and make predictions. Use decomposition to explore parts and relationships within data sets. Ask questions about available data sources, and compare and analyze test results to inform decisions, plan, and refine designs.
- **Skill progression:** Students use, organize, and analyze larger and more complex data sets to ask questions; make predictions; optimize problem statements and designs; support claims; and clearly communicate with evidence. Apply mathematical and scientific practices to analyze data, identify relationships within data, and optimize designs. Select among data visualization options and justify choices. Assess strengths and limitations of available data sources.

Instructional Considerations for Collaboration Practices

Possible **instructional approaches** to foster computational thinking practices:

1. Model strategies for breaking complex information into smaller parts. Provide opportunities to analyze and discuss the relationship among parts.
2. Support students with recognizing patterns. Model how to analyze, interpret, and display patterns to make predictions and draw conclusions.
3. Model the use of abstraction (e.g. visualizations, storyboards, flowcharts, decision trees, models, computer simulations) to simplify problems; represent information (e.g. data, patterns, processes, phenomena, systems); organize thinking; and support sense-making. Support students with creating and evaluating abstractions and their limitations.

4. Plan opportunities for students to use sequencing in problem solving, incorporate user feedback, and check for bias, accessibility, and other design criteria. Model ways to systematically test, validate, evaluate, refine, and optimize algorithmic solutions. Provide opportunities to reflect on how algorithms are used in solutions.
5. Model abstraction, pattern analysis, and decomposition. Use models to develop and test predictions. Identify limitations and benefits of models.

Instructional activities may include but are not limited to:

- **Create Artifacts**: Students could create an app, program, animation, simulations, etc. to solve a community problem or creatively express an idea.
- **Identify Patterns to Make Predictions**: Students notice repetition in sequences of numbers or parts of a process to make predictions about future events or missing components.
- **Create Abstractions:** Students create and compare when various abstractions support problem solving, such as models, visualizations, storyboards, flowcharts, decision trees, generalizations, simulations.
- **Modular Programming:** Students break down the steps needed to solve a problem then document subgoals of existing processes. Use those subprograms to simplify programming or reuse solutions for other problems.
- **Create Models:** Develop models to represent information such as patterns, relationships, inputs/outputs. Create models of systems (e.g. model networks, cybersecurity, emerging technologies) to understand how parts connect to perform a function. Students can create models to engage in systems thinking and modularization.
- **Evaluate existing models and programs:** Research technologies and evaluate outputs for bias, accessibility, reliability or other established design criteria. Students can identify applicable parts or modules of existing programs and reuse to solve different problems.
- **Reflection and Transfer:** Have students reflect on how each computational practice facilitates problem-solving and identify opportunities to transfer the practice to other situations. Support students in identifying key points in feedback.

C. Fostering Iterative Design Practices

1. **Identify, Define, and Evaluate Real-World Problems:**

- **All levels:** Students identify, define, and explore existing problems and potential solutions. Ask questions to clarify project goals and explore problems from different perspectives. Clarify success criteria, identify constraints, and uncover missing information. Explore patterns and develop generalizations about the types of problems that benefit from computational solutions.

- **Skill progression:** Students also define and analyze increasingly complex, interdisciplinary problems that can be addressed computationally. Evaluate the efficiency, ethical concerns, and feasibility of solutions. Examine how computer science and emerging technologies affect problem-solving.

2. **Plan and Design Artifacts:**

- **All levels:** Students generate ideas for new solutions incorporate ideas from peers into designs and reflect on how diverse perspectives affect plans and designs. Use tools like flowcharts, mind maps, storyboards, outlines, and decision trees to plan prototypes. Compare solutions to criteria and constraints. Use quantitative and qualitative data to choose a solution to prototype. Predict the performance and impacts of prototypes, including user needs, and accessibility.
- **Skill progression:** Students intentionally apply computational thinking and planning tools such as pseudocode with documentation, models, and decision trees to design solutions. Ask questions about data sources and outline methods for testing prototypes, including checking for errors, sources of bias, and alignment with design criteria. Evaluate and modify plans to address diverse user needs using empathy, feedback, and iterative refinement.

3. **Create, Communicate, and Document Solutions:**

- **All levels:** Students use plans to create artifacts such as algorithms, programs, and prototypes. Describe design choices and make connections to the design challenge, criteria, and constraints. Engage in giving and receiving feedback to refine solutions and enhance communication skills. Annotate their programs to explain design choices. (This known as "documenting code" in the computer science field.)
- **Skill progression:** Students also modify or remix parts of existing programs to develop their ideas, streamline designs, or add more advanced features and complexity. Provide documentation for end users that explains the functionality of artifacts. Seek input from broad audiences and intentionally apply iterative design. Develop detailed and clear comments, graphics, presentations, and demonstrations.

4. **Test and Optimize Artifacts:**

- **All levels:** Students test artifacts to ensure they meet criteria and constraints, comparing results to intended outcomes. Use computational thinking and other problem-solving strategies like trial and error to fix simple errors, debug, revise, and evaluate artifacts against design criteria. Reflect on how the iterative design and computational thinking practices facilitate program development.
- **Skill progression:** Students also employ systematic and iterative testing to identify and resolve issues and optimize program design. Anticipate and address potential errors and edge cases to improve reliability and functionality. Distinguish between syntax and logic errors. Identify strategies to improve implementation of the iterative design process.

Instructional Considerations for Collaboration Practices

Possible **instructional approaches** to foster iterative design practices:

1. Design learning experiences where students identify real-world problems and evaluate the appropriateness of using computational tools to develop solutions.

2. Provide instructional time and model strategies to support students with using an iterative process to plan the development of a computational artifact while considering key features, time and resource constraints, and user expectations. Design instructions to provide students with multiple paths to solve problems.
3. Provide instructional time for students to prototype**,** justify, and document computational processes and solutions using iterative processes. Model how to listen to differing ideas and consider various approaches and solutions.
4. Provide instructional time and model strategies for evaluating computational artifacts using systematic testing and iterative refinement to enhance performance, reliability, usability, and accessibility as outlined in the design criteria.

Instructional activities may include but are not limited to:

- **Class Debates:** Debate pros and cons of using computing technologies to solve real-world problems. Consider examples like drones monitoring the environment; AI-generated art; or personalized learning applications. Progressive examples include, using machine learning in self-driving cars to interpret road conditions and make decisions, and robots assisting in surgeries for precision and reduced recovery times.
- **Prototype and Improve:** Create simple animated stories; solve pre-existing problems; utilize coding platforms to simulate a solution; incorporate microcontrollers to create physical models. Use peer feedback to refine the design. Document changes and justifying improvements at each step.
- **Debug and Enhance:** Work with a pre-built program containing intentional errors and limited features to debug syntax and logic issues, optimize the program for performance, and enhance it with new capabilities.
- **Accessibility Upgrade:** Emphasize empathy and inclusion in design by analyzing an existing program or interface (e.g., a basic website). Evaluate it for usability and accessibility. Propose and implement iterative changes to improve the design, such as adding features like text-to-speech, adjustable font sizes, or simplified navigation.

D. Fostering Digital Literacy Practices

1. **Responsible Use Practices:**

- **All levels:** Students use technology in ways that are safe, legal, and ethical. Explore data privacy rights, data protections, terms of service and privacy policies. Implement strategies to protect their digital identity, personal data, and the data of others. Explore and ask questions about how computer science and emerging technologies work, and their benefits and risks. Consider the benefits and risks of sharing different types of information with different audiences.
- **Skill progression:** Students analyze data privacy rights, data protections, terms of service and privacy policies. Weigh tradeoffs and risks with actions and decisions involving computer science. Justify choices for responsible use of computing technology using accurate information of how current and emerging technologies work.

2. **Safeguard Well-Being of Self and Others:**

- **All levels:** Students reflect on their emotional response to the use of digital technology and identify how to use technology in ways that support personal well-being. Consider how the use of technology can impact others and make choices that benefit others and avoid harm. Identify the roles and responsibilities of humans in designing and using technologies and avoid anthropomorphizing tools like AI. Practice empathy and engage in positive, respectful online practices as an upstander.
- **Skill progression:** Students also reflect on their choices about technology use and assess how different technology support learning goals and the well-being of self and others. Practice setting boundaries for online communication.

3. **Evaluate Resources and Recognize Contributions:**

- **All levels:** Students apply strategies for evaluating the accuracy and relevance of digital sources. Begin to evaluate for reliability, credibility, perspective, limitations, appropriateness, and accessibility of digital sources. Keep track of sources of information and give credit to the creators of information. Identify false or misleading information.
- **Skill progression:** Students also evaluate validity, consistency, appropriateness for needs, data bias, importance, and social and cultural context. Consider assumptions, missing information, and perspectives. Identify and give attribution to ideas that are borrowed and modified, and check for licensing permissions.

Instructional Considerations for Collaboration Practices

Possible **instructional approaches** to foster digital literacy practices:

1. Model how to use technology in ways that are safe, legal, and ethical. Model how to make decisions about data privacy and information sharing that protect individual and peer identify and digital footprint. Incorporate learning activities like discussions of digital dilemmas that help students explore different perspectives, benefits, risks, and tradeoffs.
2. Incorporate opportunities for students to reflect on possible positive and negative impacts of how they use computing technologies. Choose instructional technology that aligns with learning goals and use data on students learning to reflect on and assess the extent to which the technology is supporting learning outcomes. Provide opportunities to identify the role of humans in developing and using technology and avoid anthropomorphizing tools like AI.
3. Model strategies for how to investigate the credibility of information sources and give appropriate attributions for content created by others.

Instructional activities may include but are not limited to:

- **Source Evaluation:** Assign students articles. Have students distinguish between fact and opinion within articles and evaluate the reliability of the sources.
- **Design Thinking:** Seek user-feedback throughout the design process to gain primary source information. Compare findings to predictions to help uncover and correct assumptions.
- **Tracing:** show **(**trace) where relevant evidence supports a claim, program, or model.

- **Comparative Analyses:** Encourage students to explore ethical dilemmas, compare different approaches to data privacy and possible impacts across different time periods using evidence to support arguments.
- **Class Debates:** Organize debates where students take on roles representing different perspectives and defend their positions.
- **Digital Dilemmas:** Discuss case studies of complex topics that do not have one right answer such as the CommonSense Education digital dilemmas.

Appendix B

Grade 6 Computer Science Vocabulary

Term	Definition
Abstract Data (ADT)	Models that use a set of possible values and operations to define data.
Abstraction	A filtering process used to create a simplified representation of relevant data to identify essential details, excluding less important details.
Acceptable Use Policy (AUP)	Rules and guidelines that define safe practices and responsible use of technology.
Accessibility	Refers to the design of products, devices, services, or environments for people with disabilities. These disabilities may include visual, auditory, motor, and cognitive disabilities.
Algorithm	Finite and specified set of step-by-step instructions designed to solve a problem or perform a task
American Standard Code for Information Interchange (ASCII)	A character encoding standard that represents text in numeric form, enabling multiple data representations in computing devices.
Application Software	A type of computer program designed to perform specific tasks for users. It is different from system software, which manages hardware and basic system operations.
Artificial Intelligence (AI)	A branch of computer science focused on the research and development of algorithms and systems that simulate tasks that typically require human intelligence, such as reasoning, learning, perception, and decision-making.
Assistive Technology	Any device, software, or system that is used to increase, maintain, or improve functional capabilities of a person with a disability.
Attribute	Characteristic or quality that helps us describe and differentiate objects or data (i.e. color, size, shape, weight, position, number, or texture).
Authentication	A process used to verify a user's identity before accessing a network or computer system.

Authorization	A process used to verify a user's level of access to a computer system.
Automated Decision Making	The use of predefined algorithms or programs that enable a computing device the ability to make decisions independently based on the information it receives.
Bias	Prejudice in favor of or against one thing, person, or group compared with another.
Binary	A number system that uses two digits, 0 and 1.
Block-Based Programming	A visual drag and drop programming tool that users can use to create programs using command blocks.
Bubble Sort Algorithms	Algorithms that compare consecutive items in a list and will switch the items that are in the wrong order.
Bug	An error or mistake in a program that results in unintended outcomes such as an incorrect response or crash.
Cipher	A secret or disguised way of writing; a code.
Classify	Is to arrange or organize a set of objects according to a predetermined category or attribute (a quality or characteristic).
Client	A computer or software application that retrieves and uses information, resources, or services from another device over a network.
Cloud Computing	Storing and accessing information on the internet.
Code	Any set of instructions expressed in a programming language.
Code Tracing	The practice of reading and analyzing a section of code and identifying the values of critical variables in the algorithm.
Collecting	Gathering the appropriate type of data needed.
Compound (or composite) Data	Data that combines two or more primitive data, such as a record that contains multiple variables. Sometimes called Composite Data.
Computational Artifacts	Any creation made by a human using a computing device. Can include but are not limited to prototypes, programs, planning documents, animations, or abstractions (e.g. visualizations, storyboards, flowcharts, decision trees, models, computer simulations).
Computational Thinking	A logical and systematic problem-solving process that uses decomposition, pattern recognition, abstraction, and algorithm thinking to foster creativity and develop solutions.

Computer	An electronic computing device that processes, stores, and retrieves data and capable of executing a wide range of tasks, from basic calculations to complex data processing.
Computer Science	The study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society.
Computer System	Integrated group of hardware and software that work together to store, process, and manage data.
Computing Device	Electronic tool or machine designed to receive, store, and process data to perform tasks.
Computing Technologies	Broad range of devices and tools that help us process information and perform tasks using computers and software.
Conditional Control Structures	Conditional logic (e.g., if-else statements) to make decisions within a computer program.
Conditions	A set of operators or conditions such as "if" or "and" that gives a program a path to follow when executing a programmed task.
Control Structures	The parts of a program that specify the order in which instructions are executed. Control structures can analyze variables within a program code.
Cookies	Small text files that track a user's browsing history on a website along with information about their geographic location, browsing software, and operating system.
Crash	When computer software, hardware, or programming stops functioning unexpectedly due to damaged parts, faulty coding, or a virus.
Cyberattack	An attack that targets computing devices, networks, software, or users to disrupt operations, gain unauthorized access, or steal, alter, or destroy data.
Cybersecurity	Protection of data and information on networks and computing devices from unauthorized access, attacks, damage, or theft.
Data	Individual pieces of information about people, things, or events that can be processed, stored, and analyzed by computing devices.
Data Center	Facilities that house servers and store vast amounts of information. They keep the Internet running by providing the physical space and equipment needed to store and process data.

Data Cycle	Process of formulating questions to be explored with data, collecting or acquiring data, organizing and representing data, and analyzing and communicating results.
Data Visualization	The representation of data through use of common graphics, such as charts, plots, infographics and even animations to make complex data more accessible and understandable.
Database	Using online databases (e.g., academic, scientific) to find structured data and studies.
Debug	Process of identifying, isolating, and fixing errors (often referred to as "bugs") in a set of instructions, code, or system. This can also include hardware and software.
Decision Tree	A mathematical model that uses numerous questions and answers to analyze and classify data before predicting a result. Decision Tree machine learning is a supervised approach that uses a defined set of values to simplify data into a tree-like structure to answer a question or make a prediction.
Decomposition	Process of breaking down a problem, process, or task into smaller, more manageable components.
Decryption	The process of converting encrypted data back into its original, readable form.
Design	Creation of a plan or prototype of a proposed solution.
Design Document	A detailed plan that outlines the structure, features, and implementation strategy of a project. It serves as a blueprint, providing clear specifications, goals, and guidelines for developers, designers, and stakeholders. Design documents often include diagrams, technical requirements, workflows, and rationale to ensure a shared understanding of the project's direction and execution.
Digital Citizenship	The rights, responsibilities, and opportunities of living, learning, and working in a interconnected digital world. This promotes responsible and ethical behavior in digital environments, including understanding data privacy, security, and the impact of digital actions.
Digital Literacy	The ability to use technology effectively and responsibly to access, evaluate, create, and communicate information.
Diverse	Variety of different elements, qualities, or characteristics.
Documentation	The act of annotating the program to explain the purpose of each section of code, also known as commenting code.
Domain Name System (DNS)	A “phonebook” for the Internet, translating website names (like google.com) into IP addresses that computers use to locate each other.
Email	A program used to send and receive messages over the Internet for online communication.

Email Server	Software applications that manage email exchange between users. They store, send, and receive email messages, enabling online communication.
Emerging Technology	New or developing technologies that are transforming.
Encode	Convert (information or an instruction) into a particular form.
Encryption	The conversion of electronic data into another form, called ciphertext, which cannot be easily understood by anyone except authorized parties.
Ethernet Cable	A type of network cable used to connect devices to create a local area network.
Evaluation	Assessment process that reviews test results and feedback to determine a design or product's effectiveness and identify necessary changes for improvement.
Expression	In a programming language, a combination of explicit values, constants, variables, operators, and functions interpreted according to the particular rules of precedence and of association which computes and then produces (returns, in a stateful environment) another value.
Extraction	Process of identifying, isolating, or deriving meaningful information.
Firewall	A network security system with rules to control incoming and outgoing traffic, providing security by blocking potentially harmful traffic from reaching a network.
Flowchart	A diagram that shows the steps in a process using shapes and arrows. It helps the user visualize how things happen in order.
Function	Like variables, except instead of storing data they store lines of code. Help to simplify the programming process and make code more readable.
Hardware	The physical components of a computing device that you can touch, such as the processor, memory, keyboard, and display.
Hypertext Transfer Protocol Secure (HTTPS)	A secure protocol used by a web browser application to transmit information from a website to a user.
Implementation	The development or execution of a functional prototype, program, or product.
Information	Facts provided or learned about something or someone.

Input	Data that is entered or received by a computing device for processing.
Intellectual Property Rights (IPR)	Legal protections granted to creators and inventors for their original works, designs, and inventions.
Internet	A global network of interconnected computing devices that allows devices to share information and resources.
Interpreter	A type of computer program that executes code line by line, translating it from a high-level programming language into machine code at runtime.
Iteration	Repeated actions.
Key	A distinct identifier used to differentiate data elements within a set.
Large Datasets	Collection of large amounts of data that require specialized tools or methods to store, process, and analyze.
Linear Regression	A mathematical model that analyzes linear relationships and variables to predict a result. This is a supervised learning method that uses labeled data to map points in an effort to predict future results.
Link	The connection between two different nodes that represent the data connection.
List	A data structure that stores an ordered collection of elements, which can be of any type (numbers, strings, objects, etc.).
Local Area Network (LAN)	A computer network that covers a confined area, such as an office building, university, or home.
Logic Errors	An error that occurs when a program is executed and produces incorrect or unintended output without causing the program to crash or display an error message.
Loop	A set of instructions that are repeated until a specified condition is met, or a predetermined number of repetitions has occurred.
Machine Learning	A process that occurs when computers learn from examples instead of following exact instructions. Examples of machine learning include supervised learning, unsupervised learning, and reinforcement learning.
Malware	Malware is malicious software that can steal data, corrupt files, disrupt services, and/or damage networks and computing systems.
Mathematical Model	A way for computers to predict a result or model the real world by using a set of given mathematical rules and data.

Media Access Control (MAC) Address	A unique hardware identifier assigned to each device on a network. The MAC address is embedded in the network card during manufacturing and cannot be altered.
Memory	The physical storage in computing devices where data is processed and instructions for processing are stored. Memory types include RAM (Random Access Memory), ROM (Read-Only Memory), and secondary storage like hard drives, removable drives, and cloud storage.
Merge Sort Algorithms	Algorithms that split data lists into halves called sublists repeatedly until there is only one item in each sublist.
Models	Simplified representation or abstraction of a concept, object, system, or process.
Modem	A device that connects a home or local network to the Internet. They convert data from digital form (used by computers) to a form that can travel over phone lines, cable lines, or fiber optics, allowing devices to access the Internet.
Modularity	A characteristic in programming where subcomponents (modules) with specific functions are identified and incorporated into the code or program.
Naming Convention	Guidelines for the use of descriptive names for variables, functions, and classes.
Nested Conditional Control Structures	Conditional statements placed inside the body of another conditional statement.
Network	Two or more computers that are connected together to communicate and share information.
Neural Network	A specific method of Artificial Intelligence (AI) that uses a network of computers to process data and produce an output in a way that imitates the human brain.
Node	Node is a computing device that is connected to a network and is able to communicate with other devices.
Non-Numeric Data	Refers to data that involves categories, qualities, or descriptions rather than numbers such as name, address, and favorite color.
Numeric Data	Refers to data that involves numbers and can be counted, measured, or quantified such as age, weight, or height.
Operating System (OS)	The software that manages hardware resources and provides a user interface. It enables the computer to run programs and ensures everything works together smoothly.
Operator	A symbol that establishes a relationship between two values. Common types include logical (AND, OR, NOT), relational (=, <, ≤, >, ≥), and arithmetic (+, -, ÷, ×) operators.

Output	Data or information produced by a computing device after processing input.
Packet	Packets are smaller pieces of data that can be sent across networks.
Parameter	Parameters are coding structures that identify the values that will be transferred when called.
Password	A secret word or phrase used to protect devices and information from unauthorized access.
Pattern Analysis	Process of identifying commonalities, differences, and predictable relationships within data to understand, interpret, and make predictions.
Pattern Recognition	Ability to identify commonalities, similarities, or differences in recurring elements.
Patterns	A repeated sequence or behavior that is observable in data, objects, or events.
Personal Identifiable Information (PII)	Any data that can be used to identify a specific individual. It includes both direct and indirect identifiers that, alone or combined, can reveal a person's identity.
Personal Information	Data or information about a person that relates to their identity, characteristics, or activities.
Phishing	Deceptive online attack where scammers pretend to be a trusted source and trick individuals to share personal identifiable information.
Pixel	Picture element or tiny square of color that, when combined with other pixels, comprises a larger image.
Plain Language	A description of the steps and logic in simple terms that anyone can understand through the use of familiar analogies, real-life examples, and simple terms.
Prediction	An educated guess about what will happen next, based on current facts or what is already known.
Primitive Data	Basic forms of data that can store single values.
Probeware	Scientific equipment that combines sensors (hardware) with software to collect scientific data (e.g. light, temperature, distance, motion).
Problem Definition	Clearly identifying the problem or challenge that needs to be solved.
Procedure	Broadly used to refer to a process, which may include a method, function, subroutine, or module, depending on the programming language.
Processing Speed	How fast a computer can take in information, think, and complete tasks.

Program	The implementation of an algorithm (set of instructions) translated into a programming language that a computer can follow and execute to perform a specific task.
Programming Language	Formal system of communication used to write instructions that a computer can execute. It consists of syntax (rules for structuring code) and semantics (meaning of the instructions). Programming languages allow developers to create software, automate tasks, and control hardware.
Proprietary Software	Software that is owned by an individual, company, or organization and is subject to restrictions on its use, modification, and distribution. The source code is typically not available to the public, meaning users cannot inspect, modify, or freely share it.
Protocols	Agreed upon rules that define how messages between computers are sent. They determine how quickly and securely information is transmitted across networks and the Internet, as well as how to handle errors in transmission.
Pseudocode	An algorithm written in plain language instead of a programming language.
Public Information	Information that is okay to share with anyone and is typically available for everyone to see.
Quick Sort Algorithms	Algorithms that can quickly and efficiently organize data in ascending or descending order by splitting the data into smaller pieces for processing.
Ransomware	A type of malicious software (malware) that encrypts a victim's data or locks them out of their system, demanding payment (a ransom) to restore access. It is commonly spread through phishing emails, malicious attachments, or software vulnerabilities.
Reinforcement Learning	A type of machine learning where a computer learns through trial and error. The computer receives feedback and adjusts its behavior to improve performance.
Reusability	A characteristic in programming where a subcomponent of a program has a clear, well-defined purpose that makes it possible to use it repeatedly in different parts of the program.
Robotics	Using robots to perform repetitive tasks with precision.
Router	A device that directs data from one network to another.
Routine	A series of activities or tasks that someone does regularly. A routine is like a special task or a set of instructions that a computer follows to do something over and over again. It is a way of organizing work so that the computer knows exactly what to do without having to be told every single time.

Screen Time	Time spent on a computing device.
Search Algorithm	A program or set of instructions that allows a user to find specific information with a defined set of data. Its role is to allow a user to search for the desired information.
Search Engines	A program that enables users to find information on the Internet.
Sensor	A computing component that detects, collects, or measure data that would otherwise be difficult to gather manually.
Sequence	The specific order in which instructions or steps are executed in an algorithm or program.
Server	A computer or software application that provides information, resources, or services to clients over a network.
Simulation	Replicating the behavior of a real-world process or system over a period of time.
Social Engineering	The process of bypassing computer security and gaining access to a device or system by tricking users into revealing passwords or other secure information.
Social Media	An application or website that an individual uses to communicate with other individuals in their community or around the world.
Software	A set of instructions that tells the computer how to act and respond but cannot be seen or touched.
Software Applications	Programs designed to perform specific tasks for the user, such as word processing, web browsing, gaming, or photo editing. These applications run on the operating system and make the device useful for various functions.
Sort	Used to compare a set of objects in order to find similarities and differences, so that they may be arranged and organized.
Spoofing	A method of copying, mimicking, or pretending to be a trustworthy source so that bad actors can use it for malicious purposes.
Spyware	A type of malware designed to secretly monitor and collect information about a user's activities without their knowledge or consent.
Storage	Location where data, programs, and files are kept permanently (until deleted).
Supervised Learning	A method of machine learning that involves a computer using large amounts of labeled datasets to recognize patterns, classify data, and make predictions.

Switch	Devices that act as a bridge within a network and connect multiple devices (like computers, printers, and routers) on a LAN (local area network) network allowing them to send and receive data to communicate directly with each other.
Syntax	Rules or structure of a programming language.
Syntax Errors	An error caused by a mistake in the rules or structure of a programming language, such as missing parentheses, commas, or incorrect keywords.
Table	A structured format to organize and record information in rows and columns.
Testing	The process of evaluating a program or system to assess its results and outputs, ensuring accuracy, performance, and reliability, while identifying errors.
Topology	The physical and logical configuration of a network; the arrangement of a network, including its nodes and connecting links.
Training Data	A collection of labeled or unlabeled data used to teach a machine learning model how to recognize patterns, make predictions, or perform tasks.
Translation	Conversion of symbolic representations or programming language into programming language.
Transmission Control Protocol/Internet Protocol (TCP/IP)	A framework of rules that allow a device to connect to and communicate with other devices on a network.
Trends	Are long-term directions or movements in data or behavior that indicate a general tendency or shift over time.
Troubleshoot	Process used to diagnose why a system or process is not working as expected and systematically testing solutions to resolve the issue.
Two-Factor Authentication (2FA)	Security mechanism that requires two types of credentials to verify authorization.
Unauthorized Access	Information is accessed without the permission of the owner.
Unsupervised Learning	When a computer learns to find groups or patterns without anyone telling it what's right or wrong.
User Input	Data or information that a user provides to a computer program during its execution (2024). Data that is taken in by a computer for processing (2017).

Username	A unique name that people use to log into a device or online account. It is like a nickname that helps the computer recognize who is logging in.
Variables	A programming element that is a named storage location in memory that holds a value, which can be modified during the execution of a program.
Version Control	A process in software design and programming where changes are tracked by documenting the improvements and incrementally changing the version number identifier.
Virtual Private Network (VPN)	A secure and encrypted network connection that creates a direct connection to a remote router, masks or hides the connection information, and encrypts the data that it transmits.
Virus	Malicious software (or malware) designed to spread from one computer to another, often without the user's knowledge.
Visualizations	Refer to graphical representations of data or information that help users understand patterns, trends, and relationships more effectively. Visualizations make complex data more accessible, interpretable, and actionable.
Web Browser	Software programs, like Chrome, Firefox, or Safari, that allow users to access and interact with websites and online content.
Webpage	A single document on the internet that is accessed through a web browser.
Websites	A collection of webpages on the Internet that people can visit using a web browser.
Wi-Fi	The device that allows computing devices to access the Internet without being connected to physical cables within a specific area using radio waves to send and receive data.
Workstation	A type of node that is typically a user's computer.
Worms	A type of malware designed to replicate itself and spread across computers or networks without needing to attach to a host program or file.
World Wide Web (WWW)	A system of interconnected web pages that users can access through the internet.