



VIRGINIA DEPARTMENT OF
EDUCATION

Copyright © 2025

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

All rights reserved. Reproduction of these materials for instructional purposes in public school classrooms in Virginia is permitted.

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

NOTICE

The Virginia Department of Education does not unlawfully discriminate on the basis of race, color, sex, national origin, age, or disability in employment or in its educational programs or services.

Table Contents

2024 Computer Science <i>Standards of Learning</i>	3
Introduction	3
Foundational Principles	3
Third Grade: 2024 Computer Science <i>Standards of Learning</i>	5
Algorithms and Programming (AP)	5
Computing Systems (CSY)	6
Cybersecurity (CYB)	6
Data and Analysis (DA)	7
Impacts of Computing (IC)	7
Networks and the Internet (NI)	8
Computer Science Instructional Guide Framework	10
Grade 3: Computer Science Instructional Guide	11
Algorithms and Programming (AP)	11
Computing Systems (CSY)	27
Cybersecurity (CYB)	36
Data and Analysis (DA)	46
Impacts of Computing (IC)	59
Networks and the Internet (NI)	73
Appendix A	77
K-5 Computer Science Skills and Practices Continuum.....	77
Appendix B	86
Grade 3 Computer Science Vocabulary	86

2024 COMPUTER SCIENCE *STANDARDS OF LEARNING*

Introduction

Virginia’s Computer Science *Standards of Learning* 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.

Disclaimer: The Virginia Department of Education (VDOE) does not endorse or recommend any commercial products, services, or platforms. Any trademarks, logos, or images displayed in this instructional guide are used solely for educational and illustrative purposes to support conceptual understanding. Their inclusion does not constitute an endorsement by the VDOE of the referenced products, services, companies, or organizations.

Third Grade: 2024 Computer Science *Standards of Learning*

In Third Grade, students delve into computational thinking, creating algorithms to tackle problems with a block-based programming language and the iterative design process. They develop skills in strategizing, constructing, executing, and testing algorithms incorporating loops, events, and conditional control structures. Students will create and evaluate data representations, using computing devices to model physical objects or processes, and assessing the societal impacts of widespread computing technology use. Students gain insights into inner workings of computing devices within a system, employing accurate terminology to address issues with malfunctioning systems. They adopt secure practices for protecting private information, understanding the correlation between passwords and security risks. Additionally, students apply the data cycle, acquiring skills to collect, store, and organize data for trend evaluation and pattern identification. Students will familiarize themselves with computing technology careers, understanding societal implications, and use of information created by others with proper permissions.

Algorithms and Programming (AP)

3.AP.1 The student will apply computational thinking to design algorithms to extend patterns, processes, or components of a problem.

- a. Identify a pattern in an algorithm, process, or a problem.
- b. Decompose a problem or task into a subset of smaller problems.
- c. Design an algorithm to extend either a pattern, process, or component of a problem.

3.AP.2 The student will plan and implement algorithms that consist of events and conditional control structures using a block-based programming language.

- a. Describe the concept of a conditional control structure.
- b. Create a design document to plan an algorithm using plain language, pseudocode, or diagrams.

3.AP.3 The student will use the iterative design process to create, test, and debug programs containing events, loops, and conditional structures in a block-based programming tool.

- a. Create and test programs that consist of events, loops, and conditional structures.
- b. Analyze and describe program results to assess validity of outcome.
- c. Revise and improve programs to resolve errors or produce desired outcome.

Computing Systems (CSY)

3.CSY.1 The student will model how computing devices within a computing system work.

- a. Describe the role of a processor in a computing system.
- b. Explain the relationship between the inputs, processors, and outputs.
- c. Discuss various types of input data a computer can accept and use.
- d. Model how a computing system works including input and output, processors, and sensors.

3.CSY.2 The student will use accurate terminology when troubleshooting problems when a computing system is not working as expected.

- a. Identify common troubleshooting strategies used to address a variety of hardware and software problems.
- b. Explain and apply troubleshooting strategies related to simple hardware and software problems.

Cybersecurity (CYB)

3.CYB.1 The student will apply safe practices to protect private information.

- a. Identify and distinguish personal information that should be private.
- b. Describe the importance of using a strong password.
- c. Create and use strong passwords to protect private information.

3.CYB.2 The student will identify the relationship between passwords and security risk.

- a. Describe how authentication and authorization protect private information.
- b. Identify multiple authentication methods.
- c. Discuss the security risk posed by not having a strong password.

3.CYB.3 The student will define and explain cybersecurity.

- a. Define cybersecurity.
- b. Research and identify problems and consequences related to inappropriate use of computing devices and networks.
- c. Model safe and responsible behaviors when using computing technologies and online communication.

Data and Analysis (DA)

3.DA.1 The student will gather, store, and organize data to evaluate trends and identify patterns using a computing device.

- a. Formulate questions that require the collection or acquisition of data.
- b. Gather, organize, sort, and store data.
- c. Examine a labeled dataset to identify potential problems within the data.
- d. Discuss how data discrepancies or problems impact predictions and results.
- e. Draw conclusions and make predictions based on observed data.

3.DA.2 The student will create and evaluate data representations and conclusions.

- a. Create charts and graphs based on data collection.
- b. Analyze data to identify patterns, draw conclusions, and make predictions.

3.DA.3 The student will create models that can represent a physical object or process.

- a. Create a model to represent a physical object or process.
- b. Identify how computing devices are used to create models.
- c. Discuss the advantages and disadvantages of using computing devices to create models.

Impacts of Computing (IC)

3.IC.1 The student will identify and examine the positive and negative impacts of the prevalence of computing technologies.

- a. Identify computing technologies that have changed the world.
- b. Examine and explain how computing technologies influence and are influenced by culture.
- c. Identify social and ethical issues related to the use of computing technologies.

3.IC.2 The student will discuss and describe strategies to manage screen time.

- a. Define and describe screen time.
- b. Explain the importance of responsible screen time management.

- c. Discuss how screen time choices affect one's personal health and interactions with others.

3.IC.3 The student will identify and describe computing technology careers and their impact on society.

- a. Research computing technology careers.
- b. Describe the impact careers in computing technology have on society.

3.IC.4 The student will demonstrate how to use information created by others with permission.

- a. Discuss copyright, piracy, and plagiarism.
- b. Demonstrate how to use information created by others.

Networks and the Internet (NI)

3.NI.1 The student will describe computing networks.

- a. Differentiate between a network and the Internet.
- b. Identify the components of a computing network.
- c. Describe how a computing device connects to a network.
- d. Identify ways networks are used to transmit information.

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.

Grade 3: Computer Science Instructional Guide

In Grade 3 students delve into computational thinking, creating algorithms to tackle problems with a block-based programming language and the iterative design process. They develop skills in strategizing, constructing, executing, and testing algorithms incorporating loops, events, and conditional control structures. Students will create and evaluate data representations, using computing devices to model physical objects or processes, and assessing the societal impacts of widespread computing technology use. Students gain insights into the inner workings of computing devices within a system, employing accurate terminology to address issues with malfunctioning systems. They adopt secure practices for protecting private information, understanding the correlation between passwords and security risks. Additionally, students apply the data cycle, acquiring skills to collect, store, and organize data for trend evaluation and pattern identification. Students will familiarize themselves with computing technology careers, understanding societal implications, and use of information created by others with proper permissions.

Algorithms and Programming (AP)

3.AP.1 The student will apply computational thinking to design algorithms to extend patterns, processes, or components of a problem.

- a. Identify a pattern in an algorithm, process, or a problem.
- b. Decompose a problem or task into a subset of smaller problems.
- c. Design an algorithm to extend either a pattern, process, or components of a problem.

Understanding the Standard

Computational thinking (CT) is a logical and systematic problem-solving process that uses decomposition, pattern recognition, abstraction, and algorithmic thinking to foster creativity and develop solutions. It is universally applicable across various fields and allows 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 the use and development of technology. Computational thinking is an integral part of Virginia's computer science standards.

Computational thinking consists of four key components: decomposition, pattern recognition, abstraction, and algorithmic thinking.

- Decomposition is the process of breaking apart a problem, process, or task into smaller, more manageable components. This involves identifying and recognizing relationships among the parts.
- Pattern recognition involves identifying commonalities, similarities, or differences in recurring elements.

- Abstraction is a filtering process. It enables one to focus on important and relevant information, while excluding or hiding irrelevant or less important details.
- Algorithmic thinking is the process of developing algorithms in a logical, systematic, and procedural way to solve problems or complete a task.

Students use computational thinking and decomposition to solve problems in their daily life. As students solve problems, they begin to see that larger problems can be broken down into a collection of smaller, more manageable tasks. This helps students develop logical thinking by understanding how individual components fit together to form a complete solution.

Students will start by breaking apart the problem or task into smaller parts or components. They will use a design document or flowchart to help guide their thinking, organize the subcomponents, and enable them to recognize how the smaller parts fit together. The design document can be written in short phrases known as pseudocode. Students will leave out any minor or unimportant details in order to focus on the larger problem. Students will apply pattern analysis to identify any patterns or repeated actions found in the problem. They will create step-by-step instructions or algorithms to extend patterns, processes, or components of their problem. Patterns, processes, and components refer to the building blocks within the program that solves the problem.

[3.AP.1a] Pattern recognition is the identification of commonalities, differences, and predictable relationships within algorithms, processes, or problems. This includes the ability to identify actions or processes that repeat, identify attributes or characteristics of an item that remains the same, or consistent changes that occur. The ability to recognize patterns within algorithms, processes, and problems is a fundamental skill in computer science. In order to identify patterns, students should look for repetition, find similarities and differences, observe trends, and utilize logic to make relational connections. It enables the development of efficient solutions, improve processes, and find better ways to solve problems.

Pattern recognition facilitates problem decomposition and is a critical component in algorithmic design and data analysis. In algorithmic design, identifying repeated sequences enables the use of loops to improve efficiency by reducing redundancy and streamlining execution. Recognizing patterns in datasets allows for accurate predictions and trend analysis, helping to derive meaningful insights from complex information

[3.AP.1b] Decomposition is the process of breaking a problem or process into manageable subcomponents. These subcomponents can be analyzed to identify and recognize relationships and interactions among the subcomponents. This process involves analyzing and defining the problem, identifying its key components, establishing a clear goal or intended solution, breaking the problem into the smaller parts and identifying patterns and relationships between the smaller parts; and solving each subcomponent based on priority.

[3.AP.1c] Repeating and increasing patterns play an important role in recognizing structured sequences. Algorithmic thinking supports the identification of patterns in data sequences or processes, enabling the creation of step-by-step instructions to extend patterns systematically. By recognizing relationships within patterns, algorithms can be designed to automate processes and predict future outcomes, enhancing efficiency and problem-solving capabilities.

- For example, students identify repeating shapes in a drawing program and write an algorithm to generate additional shapes in a systematic and predictable way.

Concepts and Connections

CONCEPTS

Computational thinking is a thought process that can be applied to solve problems in a systematic and structured way. Computational thinking consists of four key components: decomposition, pattern recognition, abstraction, and algorithmic thinking. Development of computational thinking skills enhances problem-solving abilities.

CONNECTIONS

Within the grade level/course: At this grade level, students engage in the computational thinking practices of decomposition, pattern analysis, and algorithmic thinking as they design algorithms to extend patterns, processes, or components of a problem (3.AP.1).

Vertical Progression: In Grade 2, students use computational thinking skills to identify patterns, and design algorithms to compare and contrast objects based on attributes (2.AP.1). In Grade 4, Students expand their computational thinking skills as they identify and design multiple algorithms for the purpose of comparing and contrasting them to determine which algorithm is most effective for a given task (4.AP.1).

ACROSS CONTENT AREAS

Mathematics

- **3.NS.3** The student will use mathematical reasoning and justification to represent and compare fractions (proper and improper) and mixed numbers with denominators of 2, 3, 4, 5, 6, 8, and 10), including those in context.

Science

- **3.1** The student will demonstrate an understanding of scientific and engineering practices by planning and carrying out investigations. *3.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 3.1 include: 3.2 (force).*

DIGITAL LEARNING INTEGRATION

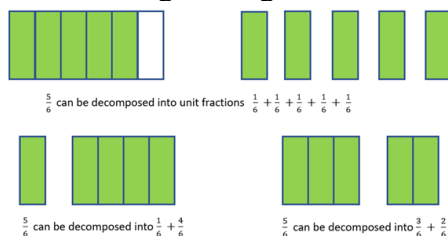
- **3-5.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
C. Break problems into component parts, extract key information, and develop descriptive models, using technologies when appropriate, to understand complex systems or facilitate problem-solving.

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.

Mathematics

- Students design an algorithm to decompose a fraction. Fractions can be decomposed in a variety of ways.



Science

- Students perform an experiment by comparing two methods for lifting a heavy object: lifting it directly upward versus using a ramp. Students decompose the task into smaller steps: measuring force, recording data, and analyzing results. Through data collection and analysis, students follow a structured procedure to collect consistent data and identify patterns that demonstrate how inclined planes enhance 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 COMPUTATIONAL THINKING PRACTICES:

1. Decompose Real-World Problems
2. Explore Common Features and Identify Patterns
3. Use 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

3.AP.2 The student will plan and implement algorithms that consist of events and conditional control structures using a block-based programming language.

- a. Describe the concept of a conditional control structure.
- b. Create a design document to plan an algorithm using plain language, pseudocode, or diagrams.

Understanding the Standard

Computational thinking is applied in planning and implementing algorithms that are used to solve problems or complete tasks. Algorithms are finite and sequential, with each step following a specific order. Algorithms can be created with or without a computing device. Students plan algorithms using a design document or flowchart that help guide their thinking. Design documents are detailed plans that outline 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. The design document or flowchart can be written in plain language, pseudocode, or simple diagrams.

A block-based programming language environment is a visual coding platform designed to help learners, especially elementary students, understand programming concepts without needing to write complex text-based code. Instead of typing instructions, users create programs by dragging and connecting graphical blocks that represent different commands, events, and control structures. Each block has a specific function (e.g., move, turn, repeat) and can be connected to other blocks to form a sequence, or algorithm, that tells the computer or app what to do.

In a block-based programming environment, events can include actions like clicking a button, pressing a key, or receiving input from sensors. Events are a fundamental concept in programming that enables programs to respond to user interactions or system triggers. They act as signals that initiate specific actions within a program, allowing dynamic and interactive behavior.

When an event occurs, it triggers a predefined sequence of code, ensuring the program reacts appropriately. Understanding events is essential for designing interactive applications, as they dictate how a program responds to external input, enhancing user experience and functionality

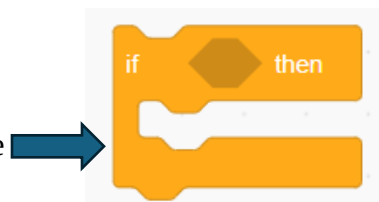
- Event is an action or something that causes a program or a certain portion of the program to run (e.g., mouse clicks on the run block).

[3.AP.2a] Conditional control structures are decision-making constructs used in programming. They consist of “if” and “if-then” statements, which provides rules that guide a computer in making decisions based on whether a specific condition is true or false. Conditional control structures use conditional logic to determine what actions the computer should take. During the execution flow of a program, the program checks whether a specific condition is true and then is instruction on what action to take based on the result. If the condition is true, the

computer performs one set instructions are implemented; if the condition is false, the program implements another set of actions or does not run at all.

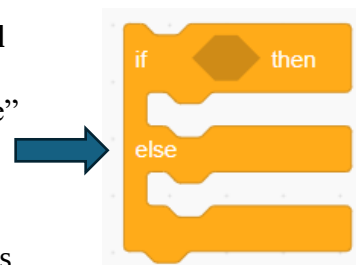
Conditional control structure framework:

“If” statements execute only when the condition is true.



Conditional control structure framework:

“If-else” statements provide different paths a program can follow when the condition is false



Source: <https://scratch.mit.edu/?scrlybrkr=eea9261a>

- **Pseudocode** example illustrates a conditional control structure.

```
IF color is green THEN
    Move car forward
END IF
```

```
If color is red THEN
    Stop car
END IF
```

The END If indicates the end of the conditional block.

[3.AP.2b] Planning is an essential part in the development of code. As students brainstorm and organize ideas through the planning process, students learn to develop and structure solutions effectively. The planning process emphasizes the ability to identify and understand the problem or task that requires a solution and the ability to design algorithms to address the problem or task. Effective planning enables systematic thinking and reduces errors during implementation. It is imperative that planning precedes the programming process (coding). It creates structure, sets goals, and organizes ideas to increase the likelihood that the program works correctly and efficiently.

Design documents are detailed plans that outline 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. In programming, design documents include plain language, pseudocode, or simple diagrams. Plain language refers to simple, everyday language used in speaking and writing. In contrast, pseudocode combines plain language and programming keywords to outline steps in an algorithm. Flowcharts are diagrams that show a sequence of steps to complete a task or solve a problem. Flowcharts use arrows and symbols to show the process or flow of actions in order. Design documents serve as a bridge between abstract thinking and concrete solutions, reinforcing computational thinking.

Concepts and Connections

CONCEPTS

Designing and implementing algorithms that incorporate conditional control structures involves applying logical decision-making processes to determine the appropriate execution flow based on specific conditions. This requires evaluating multiple inputs and potential outcomes to ensure the algorithm can dynamically respond to varying data and user interactions. Conditional statements such as 'if' and 'if-then' serve as the fundamental constructs for guiding program execution through these decision branches.

CONNECTIONS

Within the grade level/course: At this grade level, students continue building upon the block-based programming language by adding conditional control structures (3.AP.2).

Vertical Progression: In Grade 2, students planned and implemented algorithms that consisted of events and loops using a block-based programming language (2.AP.2). In Grade 4, students build upon their block-based programming language through the use of loops, variables, and user-input to their events and conditional control structures (4.AP.2).

ACROSS CONTENT ALIGNMENT

English

- **3.R.1A** Identify a topic and generate questions that explore the topic.
- **3.R.1D** Organize and share information orally, in writing, or through visual display.
- **3.W.2ii** Developing, selecting, and organizing ideas relevant to topic, audience, purpose, and genre.

History and Social Science

- **3.1g** The student will apply history and social science skills to define citizenship and explain the rights and responsibilities of United States citizenship by g. serving on a jury.
- **3.4** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Egypt.
- **3.5** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient China.
- **3.6** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Greece.
- **3.7** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Rome.
- **3.8** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of the ancient empire of Mali.

Mathematics

- **3.PFA.1** The student will identify, describe, extend, and create increasing and decreasing patterns (limited to addition and subtraction of whole numbers), including those in context, using various representations.
- **3.MG.4** The student will identify, describe, classify, compare, combine, and subdivide polygons.
- **3.CE.2** The student will recall with automaticity multiplication and division facts through 10×10 ; and represent, solve, and justify solutions to single-step contextual problems using multiplication and division with whole numbers.

Science

- **3.1** The student will demonstrate an understanding of scientific and engineering practices by planning and carrying out investigations. *3.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 3.1 include: 3.7 (water cycle).*

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 write a persuasive essay explaining why the water cycle is essential to life on Earth. Then, they will use a design document to plan and create a simple interactive presentation or digital story using a block-based coding platform. In the program, clicking on different stages of the water cycle (evaporation, condensation, precipitation, and collection) will trigger animations or messages. Conditional statements (e.g., "If there is no evaporation, then...") will guide users through various “what if” scenarios to explore the impact of each stage.

History and Social Science

- Students examine classroom or community rules and identify examples of conditional logic (e.g., “If you do not follow rules, then there are consequences”). They will then use a design document to plan an algorithm that includes at least one conditional control structure. Using a block-based coding platform (e.g., Scratch), students will create an interactive project that models responsible citizenship. Their program will include user-driven scenarios

Mathematics

- Students design an interactive digital number pattern game using a block-based programming tool. The game will require players to identify or extend number patterns. Students will use a design document to outline the game’s number pattern rules, user inputs, and expected outcomes. The final program must include conditional statements (e.g., “If the number is correct, then go to the next level”) and events to control the flow of the game based on the user’s input.

Science

- Students collect and record data on plant growth under different light conditions (e.g., sunlight, shade, artificial light). After analyzing the data for patterns and trends over time, students will use a design document to outline the steps of an algorithm that predicts future plant growth based on light exposure.

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 Identify Patterns
3. Use 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

3.AP.3 The student will use the iterative design process to create, test, and debug programs containing events, loops, and conditional structures in a block-based programming tool.

- a. Create and test programs that consist of events, loops, and conditional structures.
- b. Analyze and describe program results to assess validity of outcome.
- c. Revise and improve programs to resolve errors or produce desired outcome.

Understanding the Standard

Programming serves as a tool for problem-solving, automation, and data analysis across multiple industries. It enables the development of applications for education, scientific research, healthcare, cybersecurity, and business optimization. Programming can be used to solve complex problems and create diverse products, fostering creativity and innovation across various fields.

To ensure quality and functionality, programmers continually review and refine their work. This is a fundamental skill that should be introduced early and applied across multiple disciplines. The iterative design process is a step-by-step approach to creating and improving something. It involves making a first version, testing it, getting feedback, and making changes to make it better. Each cycle of improvement helps refine the design and build more reliable results. This process encourages problem-solving, creativity, and learning from mistakes.

This iterative process involves systematically evaluating, refining, and revisiting algorithms to guarantee its accuracy and effectiveness. It is the primary method for verifying whether algorithms correctly represent the necessary steps to accomplish a task. This process applies to both computer-based programming and without computing devices (unplugged activities).

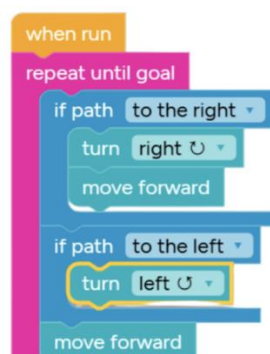
When an algorithm fails to perform as expected, students should identify and implement necessary modifications. These adjustments involve adding, removing, re-arranging, or altering steps to achieve the desired outcome. Through this continuous cycle of testing and refinement, students cultivate a deeper understanding of problem-solving and design thinking, strengthening their analytical and computational skills.

[3.AP.3a] A program is a complete set of instructions that a computer follows to perform a specific task. Code refers to the individual instructions or statements written in a programming language that make up a program. Programming is the process of writing, organizing, testing, and improving code to create a working program. Events are a fundamental concept in programming that enables programs to respond to user interactions or system triggers. They act as signals that initiate specific actions within a program, allowing dynamic and interactive behavior.

In a block-based programming environment, events can include actions like clicking a button, pressing a key, or receiving input from sensors. When an event occurs, it triggers a predefined sequence of code, ensuring the program reacts appropriately. Understanding events is essential for designing interactive applications, as they dictate how a program responds to external inputs, enhancing user experience and functionality.

- Event is an action or something that causes all of a program or a certain portion of the program to run (e.g., mouse clicks on the run block).

Example of what an event looks like in block-based programming with a loop and conditional structures.



[Image Source](#)
[Code.org](#)

Here, the event block is the orange "when run" block. When the student clicks on the run button, the program starts. The loop will repeat until the character reaches the goal. The loop contains two conditional structures where the character performs differently if the path is to the right or to the left.

Loops are a fundamental concept in programming, enabling a set of instructions to be repeated multiple times until a specific condition is met or for a predetermined number of cycles. They help automate repetitive tasks, improve efficiency, and reduce redundancy in code. A loop follows a structured sequence, executing a defined set of actions repeatedly until its stopping condition is satisfied. By recognizing repeating patterns in an algorithm, loops allow programmers to replace manual repetition with streamlined, organized code. In block-based programming, loops work alongside events to create dynamic and responsive programs. Applying an understanding of loops helps modify and enhance a program, ensuring it runs efficiently while maintaining a clear and structured design.

Tracing is the process of following each step of an algorithm or program to understand what it does before you run it. It enables you to see how a program works, what the inputs are, and what the outputs will be. Tracing allows you to anticipate the results and catch possible mistakes. It's an important part of debugging and making sure the code will work as expected. Testing, on the other hand, is when a program is executed and one can see if the program works the way it was intended. Both tracing and testing are important for finding and fixing mistakes, so the program works correctly and does what it's supposed to do.

[3.AP.3b] When a program is executed, its results or output are analyzed to determine the underlying structure and logic of the code. By examining how different instructions interact, the execution flow can be traced, revealing how the program processes data and produces outcomes. This analysis allows students to identify relationships between various components of the program, explain the program's behavior, and refine its functionality. Understanding the execution flow is essential for debugging and ensuring the program operates as intended.

[3.AP.3c] When a program does not produce the expected results, debugging is performed. Debugging is a process of identifying and resolving errors in the code. Debugging may involve adding, removing, rearranging, or adjusting specific steps to achieve the intended outcome. This process often requires analyzing error messages, tracing the execution flow, and testing different solutions to isolate the issue. Maintaining clean and well-structured code improves readability, making it easier to troubleshoot and refine for optimal performance.

Concepts and Connections

CONCEPTS

The iterative design process involves creating, testing, and improving a program through repeated cycles. Through each iteration, developers test their program, identify bugs or errors, and make changes to improve how the program works. In block-based programming tools, students build code using visual blocks that represent different commands or actions.

CONNECTIONS

Within the grade level/course: At this grade level, students expand their use of the iterative design process to create, test, and debug programs containing events, loops, and conditional structures in a block-based programming tool (3.A.3).

Vertical Progression: In Grade 2, students' algorithms included events and loops that they created, tested, and debugged using a block-based programming tool (2.AP.3). In Grade 4, students use block-based programming to include user input as well as to create and use variables for the purpose of storing and processing data. They use the iterative design process to trace and predict the value of the included variables (4.AP.3).

ACROSS CONTENT AREAS

English

- **3.C.1A.** Participate in a range of collaborative discussions (one-on-one, in groups, and teacher0lead) on grade three topics and texts.
 - iii. Asking and responding to questions that acquire or confirm information on a topic and link their comments to the remarks of others.

Mathematics

- **3.PFA.1** The student will identify, describe, extend, and create increasing and decreasing patterns (limited to addition and subtraction of whole numbers), including those in context, using various representations.
- **3.MG.4** The student will identify, describe, classify, compare, combine, and subdivide polygons.

Science

- **3.1** The student will demonstrate an understanding of scientific and engineering practices by planning and carrying out investigations. *3.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 3.1 include: 3.2cd (simple and compound machines).*

DIGITAL LEARNING INTEGRATION

- **3-5.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. Know and use appropriate technologies in a purposeful design process for generating ideas, testing theories, creating innovative digital works, or solving authentic problems.
 - C. Use appropriate technologies to develop, test, and refine prototypes as part of a cyclical design process.

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 write short narratives and exchange narratives with peers to identify and revise grammatical, spelling, or clarity issues. This is similar to how programmers debug code to resolve errors and improve functionality.
- Students create a branching digital story using a block-based coding tool, where students test and revise their code to ensure characters respond correctly to choices (using conditionals), reinforcing the writing process as iterative and dependent on user input.

Science

- Students create an algorithm to simulate a scientific process, such as the water cycle or the life cycle of a butterfly. They will use loops to repeat actions that occur more than once in the process (e.g., “Repeat evaporation and condensation in the water cycle”). The algorithm is implemented using a block-based coding tool to show how loops help simplify repeated steps in science.

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 INTERACTIVE 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

[Back to Algorithms and Programming \(AP\)](#)

Computing Systems (CSY)

3.CSY.1 The student will model how computing devices within a computing system work.

- a. Describe the role of a processor in a computing system.
- b. Explain the relationship between the inputs, processors, and outputs.
- c. Discuss various types of input data a computer can accept and use.
- d. Model how a computing system works including input and output, processors, and sensors.

Understanding the Standard

A computing system is an electronic device designed to receive, process, store, and produce or send information. It consists of hardware and software that work together to perform tasks. Hardware refers to the physical components of a computing system that can be seen and touched. This includes a monitor or screen for displaying information, buttons to power the device on and off, as well as a mouse and keyboard for inputting data. Hardware also encompasses the microprocessor or central processing unit (CPU), the system's command center, which processes all incoming (input) and outgoing (output) data.

[3.CSY.1a] The CPU executes programming instructions by performing various operations. The CPU is often referred to as “the brain” of the computer, processing information and making decisions to ensure the system runs and is responsive to change. As the processor carries out instructions to perform tasks, it uses memory to store and retrieve data. Memory is a hardware component where data is stored for retrieval and processing.

[3.CSY.1b] The relationship between inputs, processors, and outputs is interwoven. A computing system receives input and provides output. Input is the information, or instructions entered a computer using a mouse, keyboard, microphone, touchscreens, camera and sensors.

- A sensor is a device that can input or transmit data to a computing device. Sensors can measure and collect data on light, heat, motion, and moisture

Processing uses the provided input, executes (follows) the program instructions to determine what to do with the input. Output is the result of the processed input. While output is the results of the processed input, it allows the user the ability to see, hear, or interact with the input data in a meaningful way.

Computers use binary code to communicate and process information. Binary is a special language that computers use to store and process all information, represented by two numbers: 0 and 1. Computers store and process all information using a sequence of binary numbers. These binary digits, or bits form the foundation of all computer memory.

- For example, when a user presses a key on the keyboard (input), the computer's central processing unit (CPU) reads the assigned binary code, follows the program's instructions to interpret it (process) and temporarily store information in memory if needed. After processing the CPU translates the binary code back into a readable format for the user and displays the letter or numbers on the screen or prints it on paper (output).

Examples of input-processing-output process at school:

Cafeteria Meal System:

- Input: Student input their lunch number.
- Processing: The central processing unit receives and processes the input number.
- Processing: The lunch software instructs the CPU to retrieve and process the student lunch number (input).
- Output: The monitor displays the student's breakfast and/or lunch account status and balance.

School Media Center:

- Input: A student returns a book by scanning the bar code.
- Processing: The central processing unit receives and processes the input number.
- Processing: The school media center instructs the CPU to process the number of books that can be checked out.
- Output: The monitor displays the students' checked out books.

[3.CSY.1c] Computers can accept and use various types of input data. Some categories of input data include but not limited to text input, numeric input, audio input, visual input, sensor input, and touch input. Examples would include numbers, typed words, voice commands, sensor data, music, images, gestures, mouse clicks, and more.

[3.CSY.1c] Model how a computing system receives data and then processes this data and delivers it as outputs. Students may simulate this process using age-appropriate examples, such as acting out roles (input, processor, output), using flowcharts or building simple programs or devices that demonstrate the data flow.

Concepts and Connections

CONCEPTS

Computing systems are composed of interconnected components that work together to complete tasks. It follows an important cycle known as input-process-output (I/O). The computer receives input, processes the input using various operations, and produces an output. This I/O process is the core of computing and allows computers to turn data into useful information.

CONNECTIONS

Within the grade level/course: At this grade level, students model computing systems and understand the relationship between inputs, processors, and outputs (3.CSY.1).

Vertical Progression: In Grade 1, students describe the characteristics of computing systems, including hardware, software, input, and output (2.CSY.1). In Grade 4, students model how computing systems work to accomplish a task, plus diagram and describe a computing system indicating processors, inputs, and outputs (4.CSY.1)

ACROSS CONTENT AREAS

Mathematics

- **3.CE.2** The student will recall with automaticity multiplication and division facts through 10×10 ; and represent, solve, and justify solutions to single-step contextual problems using multiplication and division with whole numbers.
- **3.PFA.1** The student will identify, describe, extend, and create increasing and decreasing patterns (limited to addition and subtraction of whole numbers), including those in context, using various representations.

DIGITAL LEARNING INTEGRATION

- **3-5.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. Communicate complex ideas clearly and effectively by creating or using a variety of digital objects such as visualizations, models, or simulations.

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 apply history and social science skills to define citizenship and explain the rights and responsibilities of U.S. citizens. They will design a digital decision-making model using a block-based programming tool that follows the input–processing–output (IPO) model: user input represents a civic choice (e.g., voting, following a law), the program processes that choice using conditionals or rules, and the output shows the resulting consequence or civic outcome.

Mathematics

- Students work with a partner to create and solve problems using input/output rules represented through tables or diagrams. Students identify patterns, develop algorithms, and explain how inputs are transformed into outputs using mathematical reasoning and computational thinking.

Science

- Students collect and organize data about different soil types and their components, then analyze how these factors affect plant growth. They model the input (soil type) and output (plant health or growth rate) relationship using tables or flowcharts and create simple algorithms to represent their findings through computational 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](#).

B. FOSTERING COMPUTATIONAL THINKING PRACTICES:

2. Explore Common Features and Identify Patterns
3. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

3.CSY.2 The student will use accurate terminology when troubleshooting problems when a computing system is not working as expected.

- a. Identify common troubleshooting strategies used to address a variety of hardware and software problems.
- b. Explain and apply troubleshooting strategies related to simple hardware and software problems.

Understanding the Standard

Students strengthen their troubleshooting skills, building on prior knowledge to resolve hardware and software problems. As such, students are expected to use accurate terminology when they are troubleshooting problems if the computing system is not working properly. When asked, students should be able to identify common troubleshooting strategies for common hardware and software problems. For instance, plug in the power cord when the battery is low, or reboot the computing device so the update can download and install. Students should also be able to explain why they are applying certain troubleshooting strategies such as plugging into a power source charges the battery and provides power for the computing system to work.

[3CSY.2a] Students are expected to explain hardware and software problems they encounter using appropriate and technical language. This may include “The computer is running slow”, “I am unable to log into my account” or “The screen is frozen”. Once students have identified problems, they should propose recommended solutions by employing common troubleshooting strategies. Troubleshooting involves identifying and correcting faults in a computing system. Since computing systems are composed of interconnected components of hardware and software, troubleshooting strategies may need to address both.

Possible troubleshooting strategies may include but not limited to restarting a device, checking the power source, checking connections, closing and reopening programs, checking settings, verifying login information, verifying Wi-Fi connectivity and trying the same task or application with a different device or program.

[3CSY.2b] Appropriate hardware troubleshooting steps include:

- If the computing device is not on, check to see if the computing device is turned on. If yes, check to see if brightness needs to be adjusted to correct a dark screen. If no, turn on the computing device. If the computing device has a separate monitor, check to see if the monitor is turned on. If yes, adjust the brightness on the monitor. If no, turn on the monitor.
- If the computing device is not working properly and it has previously, check to see if all of the power cords are plugged in and any cables are connected. If yes, restart the computer. If no, plug in the power cord and/or connect the cables.
- If the computing device is making a noise that is not aligned with its proper functioning and intended use, check to see if a key is being held down. If yes, remove the object by holding the key down. If no, restart the computer to see if it resolves the problem. Please also check to see if there is an error or warning message displayed on the screen of the computing device.

Appropriate software troubleshooting steps include:

- If the computing device is not working correctly, look to see if there is a message on the screen. If yes, try to read the message and tell your teacher what it says. If not, restart the computer.
- If the internet is not working properly, check to see if your computing device is connected to the internet. If yes, restart the computer. If no, try to log into the Wi-Fi network.
- If there is an update message, restart the computer to receive the update.

Computing Error	Potential Troubleshooting Strategy
Computing device is not working ...	Check the battery level to see if it needs charging.
Computing device is not printing ...	Check to see if my printer is selected or connected to my device.
Computing device is not responding ...	Check to see if my Wi-Fi is connected or if airplane mode is selected.
Computing device is not performing efficiently ...	Check to see if I have an update message and restart my device.
Computing device is running slowly ...	Check to see if other programs are running in the background and closing them.
Computing device screen is too dark or too bright ...	Check the brightness display and make necessary adjustments.

[3CSY.2c] The ability to clearly explain and apply troubleshooting strategies is an essential skill for students. One effective method for students to demonstrate understanding and application is by constructing and using problem-solution sentences. These sentences contain two parts. Part I identifies the problem and Part II describes the troubleshooting action taken.

- For example: “The sound wasn’t working, so I checked to see if the volume was turned up.”

If-then statements can be used to explain troubleshooting applications.

- For example: “If the printer isn’t working, then I will check if it has ink.”

Both sentence structures encourage systematic problem-solving through the application of computational thinking and connect specific problems to logical troubleshooting strategies.

Concepts and Connections

CONCEPTS

Computing devices may not always function as intended, and errors can occur during use. It is important for students to recognize that such issues are common and can often be resolved through common troubleshooting strategies. By applying these troubleshooting strategies, students learn to clearly articulate the problem encountered, identify possible causes, and take appropriate steps to resolve and articulate the solution.

CONNECTIONS

Within the grade level/course: At this grade level, students use accurate terminology when troubleshooting problems within a computing system if it is not working as expected (3.CSY.2).

Vertical Progression: In Grade 2, students use accurate terminology when troubleshooting problems within a computing system if it is not working as expected (3.CSY.2). In Grade 4, students will apply troubleshooting strategies when a computing system is not working as intended (4.CSY.2).

ACROSS CONTENT AREAS

Mathematics

- **3.CE.1** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using addition and subtraction with whole numbers where addends and minuends do not exceed 1,000.
- **3.CE.2** The student will recall with automaticity multiplication and division facts through 10×10 ; and represent, solve, and justify solutions to single-step contextual problems using multiplication and division with whole numbers.

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 explore how ancient inventors identified and fixed problems with early tools. Then, students relate this to how we use troubleshooting strategies today to fix computer issues like a frozen screen or a printer that won't work.

Mathematics

- Students construct and use problem-solution sentences to explain an error in their computation or strategy and how the error was resolved.

Science

- Students identify simple, common tasks around school or home that might be completed easier using a simple machine. They will investigate how the direction and size of a force affects the motion of an object using simple machines and apply computer science troubleshooting strategies such as systematic testing, isolating variables, and iterative refinement to improve their physical designs and/or data collection methods.

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 Identify Patterns
3. Use 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 PROCESS:

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

[Back to Computing Systems \(CSY\)](#)

Cybersecurity (CYB)

3.CYB.1 The student will apply safe practices to protect private information.

- a. Identify and distinguish personal information that should be private.
- b. Describe the importance of using a strong password.
- c. Create and use strong passwords to protect private information.

Understanding the Standard

Safe and responsible use of computing technologies is essential for protecting personal information and ensuring privacy online. Cybersecurity practices, such as password protection and visiting trustworthy sites are foundational safeguards in keeping one safe. Connecting devices to a network or the Internet provides great benefit, but care must be taken to protect personal information such as a student's name, phone number, and address.

[3.CYB.1a] Students should understand the difference between personal information that others can know and private information that is private and should remain private.

- Personal information is any information that identifies a person. This can include first name, favorite color, pet name, favorite sport, or hobbies. Personal information describes a person but does not necessarily place one at risk if the information is shared.
- Private information is personal information that contains sensitive data that must be protected as it could be used to contact, locate, or even harm a person. This can include home address, birthday, passwords, and student identification number,

Some information can be both personal and private, meaning it belongs to the user and should only be accessed or shared with someone the user knows and trusts. For example, combining your name and home address is both personal and private.

[3.CYB.1b] Passwords are used to protect devices and information from unauthorized access. Keeping passwords safe is critical to protecting personal information and maintaining online security. Secure passwords are passwords that are both strong and kept safe. A strong password is long, unique, and includes a mix of letters, numbers, and symbols. It should not contain a word personal information such as your birthday or your pet's name or words in the dictionary. Strong passwords must be kept safe, ensuring that the password is not shared or stored in an unsafe location.

When passwords are not secure unauthorized users may gain access to your accounts. They may read private messages, make changes to your settings, or misuse sensitive data. This may result in harm such as loss of important data or unauthorized purchases.

A strong password should include:

- Uppercase and lowercase letters.
- Numbers.
- Symbols.
- At least 8 characters. (Note: *This is age appropriate and should increase in length and complexity as students mature.*)

A password should not include:

- Words from a dictionary.
- The same password twice.
- Personal information.

Consider the following examples:

A strong password might be 'PxHn32s!@5'

A weak password might be '12345678'

[2.CYB.2c] One way to create a strong password is by using a passphrase, which is a combination of words that are easier to remember but still secure. Passphrases are more secure when combined with numbers and symbols, making it difficult to gain unauthorized access. Many sites have rules as to the length and composition of passwords; these rules help create stronger passwords. The practice of not sharing passwords should be emphasized in the classroom and at home.

Concepts and Connections

CONCEPTS

A strong and secure password protects private, such as your name or home address, making it harder for others to gain access. Strong passwords use a mix of letters, numbers, and symbols.

CONNECTIONS

Within the grade level/course: At this grade level, students identify and distinguish personal information that should be private, describe the importance of using a strong password, and create and use strong passwords to protect private information (3.CYB.1).

Vertical Progression: In Grade 2, students model safe and responsible behaviors when using information and computing technologies (2.CYB.1). In Grade 4, students examine the impacts of appropriate and inappropriate use of computing technologies (4.CYB.1).

ACROSS CONTENT AREAS

History and Social Science

- **3.1** The student will apply history and social science skills to define citizenship and explain the rights and responsibilities of United States citizenship.

DIGITAL LEARNING INTEGRATION

- **3-5.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.
 - B. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.
 - D. Manage their personal data to maintain digital privacy and security and are aware of data collection technology used to track their activity online.

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 can research strategies for staying safe online. Using appropriate and credible sources, they will collect, organize, and synthesize information to create an informative brochure. The brochure will use clear headings, facts, and visuals to communicate key points, reinforcing skills in research, informational writing, and audience awareness

History and Social Science

- Students will use a graphic organizer to compare and contrast rules for responsible technology use with rules established in the classroom or community, in order to analyze the purpose of rules and how they promote order, safety, and responsible citizenship.

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.CYB.2 The student will identify the relationship between passwords and security risk.

- a. Describe how authentication and authorization protect private information.
- b. Identify multiple authentication methods.
- c. Discuss the security risk posed by not having a strong password.

Understanding the Standard

[3.CYB.2a] Authentication and authorization protect private information by ensuring that only verified users can access private information. Authentication is the process of verifying identity. It can involve something a person knows (password), something a person has (Identification card), something a person does (signature), or something that describes a person (biometrics). Authorization is the process of controlling permission to access or do something. It is like receiving a hall pass to go to the library or a permission slip to go on a field trip. Authentication takes place before authorization. Together, they safeguard sensitive information from unauthorized access and cyber threats.

[3.CYB.2b] Multifactor authentication is a security method that verifies the user by using two or more ways. It is like a door that has two locks and requires two different keys to unlock the door and gain access to the house. Multifactor authentication makes it harder for unauthorized users to gain access as one needs multiple pieces of information to confirm user identity.

Consider the following multi-factor authentication example to access email account.

1. Enter credentials (username and password) to your email account. (Authentication 1)
2. After you have met the criteria for the first authentication method, a unique number is sent to your cell phone that you must enter into device or on the screen. (Authentication 2)
3. Once the unique number has been confirmed access to the email account will be granted

[3.CYB.2c] Without strong passwords, accounts and devices can be vulnerable to unauthorized access. Weak or shared passwords may result in identity theft, account misuse, and exposure to security threats.

- Identity theft occurs where another person pretends to be you.

Concepts and Connections

CONCEPTS

Authentication and authorization help keep private information safe. Authentication checks if a person is who they say they are. Authorization decides what that person is allowed to do. Using strong passwords and multifactor authentication strengthens security measures to protect private information and reduce unauthorized access.

CONNECTIONS

Within the grade level/course: At this grade level, students identify the relationship between passwords and security risk by describing how authentication and authorization protect private information (3.CYB.2).

Vertical Progression: In Grade 2, students explain the importance of using passwords to protect private information (2.CYB.2). In Grade 4, students identify and investigate best practices to safeguard information shared online and through online platforms (4.CYB.2).

English

- **3.C.1A** Participate in a range of collaborative discussions (one-on-one, in groups, and teacher-led) on grade three topics and texts.
iii. Asking and responding to questions that acquire or confirm information on a topic and link their comments to the remarks of others.

DIGITAL LEARNING INTEGRATION:

- **3-5.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.
B. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.
D. Manage their personal data to maintain digital privacy and security and are aware of data collection technology used to track their activity online.

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 analyze a list of passwords and classify them based on strength criteria (e.g., length, use of numbers, symbols). They then organize the data into a chart or graph to determine and explain the likelihood of each password being guessed, using language such as impossible, unlikely, likely, or certain to describe probability.

History and Social Science

- Students explore how ancient governments like those of Greece and Rome kept records secure and who had access to important information. Students then compare this to how today's digital systems protect private data using authentication methods such as passwords.

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.CYB.3 The student will define and explain cybersecurity.

- a. Define cybersecurity.
- b. Research and identify problems and consequences related to inappropriate use of computing devices and networks.
- c. Model safe and responsible behaviors when using computing technologies and online communication.

Understanding the Standard

Cybersecurity is essential in today's digital world as students are growing up using technology for learning, communication, and play. Understanding how to protect personal information, recognize risks, and follow safe online practices helps students become responsible digital citizens.

[3.CYB.3a] Cybersecurity is the practice of protecting systems, networks, and data from digital attacks. It involves using best practices, policies, and technologies to safeguard sensitive information. Cybersecurity helps keep personal and private information safe from hackers or people who may misuse the information.

[3.CYB.3b] Computing devices and networks are valuable tools for learning and communication. Unfortunately, these tools may be misused. This may include gaining access or sharing computing devices that are not given permission, changing or deleting someone's work without their permission, downloading or sharing music that you do not have ownership or rights, using school network or device for activities not allowed, or visiting websites that have been blocked.

[3.CYB.3c] Modeling safe and responsible behaviors when using computing technologies and online communication protects users from a variety of risks and potential harm. Safe and responsible behaviors foster respectful and ethical integrations, safeguard personal information, and ensure digital tools are used in a positive collaborative way.

Examples of appropriate use of technology.

- Practice math or reading skills on approved school web applications.
- Using approved websites for school related activities such as research about a specific animal when writing an informative text.

Examples of inappropriate use of technology.

- Visiting websites that are not approved by the school/division.
- Visiting websites to play non-educational games during instructional time.
- Sending hurtful or mean messages.

Concepts and Connections

CONCEPTS

Cybersecurity is the practice of protecting systems, networks, and data from digital attacks. Students learn to recognize risks and make safe, responsible choices online, understanding that unsafe behavior can lead to consequences.

CONNECTIONS

Within the grade level/course: At this grade level, students define and explain cybersecurity by researching and identifying problems and consequences related to inappropriate use of computing devices and networks (3.CYB.3).

Vertical Progression: In Grade 4, students examine how information is shared online and explain the importance of cybersecurity (4.CYB.3).

DIGITAL LEARNING INTEGRATION:

- **3-5.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.
 - A. Use appropriate technologies to connect with learners from a variety of backgrounds and cultures, engaging with them in ways that broaden mutual understanding and learning.
 - B. Use collaborative technologies to work with others, including peers, experts, and community members to examine issues and problems from multiple viewpoints.
- **3-5.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.
 - B. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.

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 examine how communities create and enforce rules to ensure safety and respect among citizens. They then compare these civic responsibilities to digital spaces by identifying problems that arise from inappropriate technology use (e.g., cyberbullying or misinformation) and modeling safe, respectful behaviors online.

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

[Back to Cybersecurity \(CYB\)](#)

Data and Analysis (DA)

3.DA.1 The student will gather, store, and organize data to evaluate trends and identify patterns using a computing device.

- a. Formulate questions that require the collection or acquisition of data.
- b. Gather, organize, sort, and store data.
- c. Examine a labeled dataset to identify potential problems within the data.
- d. Discuss how data discrepancies or problems impact predictions and results.
- e. Draw conclusions and make predictions based on observed data.

Understanding the Standard

The data cycle can be used to gather, store, and organize information in a structured way, making it easier to manage large amounts of data. By following the data cycle, users can review and sort the data and abstract important details. Trends can be evaluated, and patterns can be identified. As a result, better decisions can be made based on the data collected and analyzed. This process supports better decision-making based on the data collected and analyzed, helping students understand how data influences everyday choices at school, in their communities, and in the world around them.

[3.DA.1a] The data cycle includes the following steps: formulating precise questions to be explored with data; collecting or acquiring data through observations, surveys, or existing sources; organizing and representing data using tables, charts, and graphs; and analyzing and communicating results through written summaries or visual displays. Data serves as a form of evidence that supports answering questions, making predictions, and drawing conclusions. When constructing responses, it is important to reference specific data as evidence to justify claims or interpretations. Computers and internet-based tools can be used to access digital data sets, collect new data, and construct visual representations such as tables, bar graphs, line graphs, and pie charts to display and interpret findings. (Mathematics 3.PS.1 abcde)

[3.DA.1b] When collecting, gathering, and organizing data according to the data cycle, it is important to ensure that the data is accurate, relevant to the question being explored, and collected systematically. Data should be recorded consistently using clear formats to make organization and analysis easier.

Steps to gather, organize, and sort data.

- Define data goals and needs.
- Collect data.
- Clean and standardize the data by removing information not related to the goal.
- Categorize and label the data.

- Sort data based on category.
- Review and verify the data is correct.

[3.DA.1c] It is important to consider how data can be categorized or grouped to reveal patterns or trends. Data labeling is the process of adding tags or labels to raw data, such as text, video, or images, so that the computer knows the context of the data. Computers categorize these tags or labels into groups. Proper labeling, maintaining data integrity, and documenting how data is collected is critical for supporting valid analysis and reliable conclusions. Problems can occur when the data is not labeled correctly or if the data is not labeled at all.

[3.DA.1d] A data discrepancy occurs when information is incorrect, missing, or inconsistent, which can lead to misleading results. Collecting accurate, complete, and consistent data is essential to ensure predictions and conclusions are valid. This includes checking data sources, verifying entries, and correcting errors before analysis. Ensuring high-quality data improves the reliability of outcomes and reduces the risk of drawing false conclusions.

- **Inaccurate Information Example:**

Imagine you are trying to predict how many umbrellas a store should sell during a rainy week, but you accidentally use weather data that predicts sunshine all week instead of rain. Because the information is wrong, your prediction about selling umbrellas would not match what happens.

- **Missing Information Example:**

Suppose you are trying to figure out how many lunches a cafeteria should prepare, but you don't know how many students will be at school that day. Without knowing the number of students, it would be very hard to make an accurate plan.

- **Inconsistent Information Example:**

Imagine you are trying to figure out the average height of students in your class, but some students measured their height in inches and others measured in centimeters without converting the numbers. Because the measurements are not recorded in the same way, the data is inconsistent, and any results or conclusions would be confusing or incorrect.

[3.DA.1e] Technology tools, such as computers, can help us find patterns and trends in data. These tools can quickly look through large amounts of information and identify details that people might overlook. This process is called data analysis. By using data analysis, we can make better decisions and draw conclusions based on evidence.

- For example, if a store owner wants to know what people like to buy, the store owner can look at sales data to find popular items. This information helps a store decide what merchandise to sell. Computers can also use past data to predict future trends, such as estimating how many people will buy a product based on last year's sales. These predictions help businesses plan.

Questions that can be answered when analyzing data, drawing conclusions, and making predictions.

- What is the total?
- What is the average?
- What category has the most? The least?
- What can we predict from the data shown?
- Is there a pattern?
- What can we gain from analyzing the pattern?

Concepts and Connections

CONCEPTS

The data cycle can be used to evaluate trends and identify patterns. Computing devices assist in collecting, organizing, and analyzing data, and are especially useful for processing large datasets quickly and accurately. Technology enables users to more efficiently identify patterns, make predictions, and draw informed conclusions based on the data.

CONNECTIONS

Within this grade level: At this grade level, students engage with data analysis to determine what type of data is needed to solve a problem or answer a question. Students analyze data to formulate questions; collect and organize data; identify problems that impact the predictions and results of the data; and use data to make predictions and draw conclusions (3.DA.1).

Vertical progression: In Grade 2, Students collect and analyze data as well as describe possible patterns (2.DA.1). In Grade 4, students evaluate the reliability of data sources and begin using numeric values to represent non-numeric ideas for the purpose of collecting, storing, cleaning, and organizing their data to prepare it for visualizations (4.DA.1).

ACROSS CONTENT AREAS

Mathematics

- **3.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 pictographs and bar graphs.

Science

- **3.1** The student will demonstrate an understanding of scientific and engineering practices by a) interpreting, analyzing, and evaluating data; b) planning and carrying out investigations; *3.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 3.1 include: 3.2 (force, motion, and energy), 3.3 (matter), and 3.6 (soil).*

DIGITAL LEARNING INTEGRATION

- **3-5 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.
 - A. Formulate problem definitions suited for technology assisted methods such as data analysis, modeling and algorithmic thinking in exploring and finding solutions.
 - B. Collect data or identify relevant data sets, use appropriate technologies to analyze them, and represent data in various ways to facilitate problem solving and decision-making.
 - D. Understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.
- **3-5 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.
 - C. Curate information from digital sources using a variety of tools and methods to create collections of resources that demonstrate meaningful connections or conclusions.
- **3-5.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.
 - D. Explore local and global issues and use collaborative technologies to work with others to investigate solutions.

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 data by tracking how often each number appears when rolling a six-sided die. Students organize and categorize the data by grouping the outcomes based on the number rolled, then tallying how many times each number appears. Students can then create a bar graph, visually representing the frequency of each outcome.

Science

- Students conduct online research to gather data about how human activities affect ecosystems, focusing on air quality, water pollution, or habitat loss. Using the data, students will create a visual representation (e.g., chart, graph, or infographic) to identify patterns or trends. Based on their findings, students will make informed predictions about future impacts

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 Identify Patterns
3. Use 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

3.DA.2 The student will create and evaluate data representations and conclusions.

- a. Create charts and graphs based on data collection.
- b. Analyze data to identify patterns, draw conclusions, and make predictions.

Understanding the Standard

Data representations consist of tables, charts, graphs, and other visual formats to organize and display information. Creating and evaluating these representations helps students identify patterns, make comparisons, and draw accurate conclusions from data.

[3.DA.2] Charts and graphs are visual tools that represent data in a structured format, facilitating the interpretation and analysis of information. Choosing the appropriate type of data representation is crucial for accurately conveying insights and findings derived from collected and organized data. Once the data is represented, evaluating these charts and graphs is crucial to ensure they effectively communicate the intended information. This evaluation process helps confirm that the conclusions drawn from the data are accurate and meaningful, providing a solid foundation for informed decision-making and data analysis.

The ability to determine the type of data needed to answer a specific question, and then use computational tools to collect, organize, and analyze that data, is an essential skill in both academic and professional settings. The choice between using a chart or a graph depends on the nature of the data and the relationships or patterns that need to be highlighted. For example, bar charts are effective for comparing discrete quantities across categories, while line graphs are ideal for illustrating trends and changes over time.

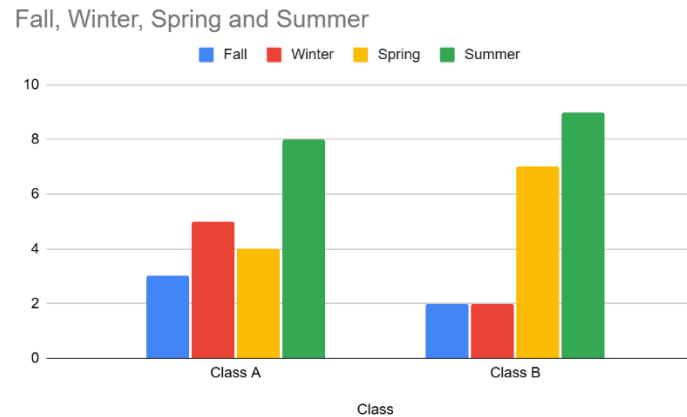
[3.DA.2a] Computers can analyze data to identify patterns, draw conclusions, and make predictions. Data serves as a set of clues that help computers recognize trends or recurring events. Professionals who work with data use advanced tools to analyze large datasets, looking for patterns that appear repeatedly, much like identifying a recurring theme in a book. For example, if data shows that sales increase every summer, a computer can use this pattern to predict future sales trends. These patterns are then used to make informed predictions about what might happen in the future. Additionally, analyzing data helps to understand the cause of certain outcomes, such as why sales dropped during a particular month. Using specialized tools, professionals can create charts and graphs that make it easier to visualize data, allowing them to clearly see patterns and draw meaningful conclusions. This process of analyzing data supports better decision-making and future planning.

[3.DA.2b] When analyzing collected and organized data, visualizations are helpful to examine and interpret the data to make predictions and draw conclusions.

Consider the following examples: Students collect data about their classmates' favorite season and record the results in a table format and organize this data in a bar graph to identify trends.

Students' Favorite Season Data Collection				
Class	Fall	Winter	Spring	Summer
Class A	3	5	4	8
Class B	2	2	7	9

Student's Favorite Season Bar Chart



Based on the bar chart, it can be concluded that summer is the favorite season for both classes. Students might also infer that summer is preferred because it coincides with a break from school. This preference could be influenced by factors such as warmer weather and the chance for outdoor activities during this break. Further analysis of this data could reveal additional insights that help explain why summer is the most popular season.

Concepts and Connections

CONCEPTS

Students use charts and graphs to visually represent data and gain a clearer understanding of patterns, trends, and relationships. They select the most appropriate type of chart or graph based on the data collected. Students evaluate these visual representations to determine whether the conclusions drawn are reasonable and meaningful for informed decision-making.

CONNECTIONS

Within this grade level: At this grade level, students create and evaluate their data representations based upon data collection and analyze it to identify patterns, draw conclusions and make predictions (3.DA.2).

Vertical progression: In Grade 2, students begin to analyze data, identify patterns and draw conclusions (2.DA.2). In Grade 4, students expand their data analysis skills by formulating questions that require data collection, recognizing and analyzing patterns and visual representations so that they can make predictions and draw conclusions based on the data (4.DA.2).

ACROSS CONTENT AREAS

Mathematics

- **3.NS.3** The student will use mathematical reasoning and justification to represent and compare fractions (proper and improper) and mixed numbers with denominators of 2, 3, 4, 5, 6, 8, and 10), including those in context.
- **3.CE.2** Students will represent multiplication and division of whole numbers through 10×10 , including in a contextual situation, using a variety of approaches and models (e.g., repeated addition/subtraction, equal-sized groups/sharing, arrays, equal jumps on a number line, using multiples to skip count). b) Use inverse relationships to write the related facts connected to a given model for multiplication and division of whole numbers through 10×10 . c) Apply strategies (e.g., place value, the properties of multiplication and/or addition) when multiplying and dividing whole numbers. d) Demonstrate fluency with multiplication facts through 10×10 by applying reasoning strategies (e.g., doubling, add-a-group, subtract-a-group, near squares, and inverse relationships). e) Represent, solve, and justify solutions to single-step contextual problems that involve multiplication and division of whole numbers through 10×10 . f) Recall with automaticity the multiplication facts through 10×10 and the corresponding division facts. g) Create an equation to represent the mathematical relationship between equivalent expressions using multiplication and/or division facts through 10×10 (e.g., $4 \times 3 = 14 - 2$, $35 \div 5 = 1 \times 7$).

DIGITAL LEARNING INTEGRATION:

- **3-5.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.
D. Understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.
- **3-3.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.

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 Sciences

- Students conduct a class survey to choose between two options (e.g., types of treats, possible field trip destinations, or a class pet). After collecting and organizing the survey data using a chart or graph, students will research and compare the basic costs associated with each option. Working in small groups, they will analyze the results to determine which choice is the most cost-effective based on student preference and price. Each group will present their findings to the class using visuals and a brief explanation of their decision-making process.

Mathematics

- Students collect classroom data using various methods (e.g., survey, poll, questionnaire) and use a computer to create a pictograph to analyze the data.

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 Identify Patterns
3. Use 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

3.DA.3 The student will create models that can represent a physical object or process.

- a. Create a model to represent a physical object or process.
- b. Identify how computing devices are used to create models.
- c. Discuss the advantages and disadvantages of using computing devices to create models.

Understanding the Standard

Scientists, computer scientists, mathematicians, and programmers construct and utilize models to better conceptualize and understand phenomena under investigation or to develop a possible solution to a proposed problem. Models can represent systems or processes that are difficult to observe directly, such as the inner workings of a computer or the flow of information across the internet. Through the use of models, students can test hypotheses, predict outcomes, and better understand complex computing systems. Modeling plays a key role in computer science by allowing the representation of real-world scenarios, supporting problem-solving and experimentation with data and behaviors. This technique is useful for visualizing numbers, natural phenomena, data trends, or system behaviors.

[3.DA.3a] Models include diagrams, physical replicas, mathematical representations, analogies, and computer simulations. Models are used to represent all or parts of a system under study; to aid in the development of questions and explanations; to generate data that can be used to make predictions; and to communicate ideas to others. A model emphasizes the most important aspects of a system, helping students focus on key data. Through abstraction, unnecessary details are removed to simplify the system and highlight the essential elements. This process makes it easier to understand the system and focus on what is most relevant. When creating a model, students should prioritize key information that will help solve a problem or answer a question.

Consider the following examples of models that represent a physical object:

- 3D Printing a Car Part: Students can design a digital model of a car part before it is printed in real life.
- Virtual Solar System: A computer simulation that lets students explore planetary orbits.
- Virtual Lego Block Model: Using code to create a visual representation of a Lego structure before building it physically.

Consider the following examples of models that represent a process:

- Traffic Light Simulation: A program that models how traffic lights change based on timing or sensor input.
- Ecosystem Simulation: A digital model of how animals and plants interact in a food chain.
- Recipe Algorithm: A step-by-step program that models the cooking process, turning it into a clear sequence of operations.

[3.DA.3b] Computer models are frequently used to help students learn. Students are able to do experiments on the computer virtually without having to use real materials or worry about accidents occurring in the classroom. Virtual experiments or simulations helps students learn about

the world around them. They can also change variables in the model and see how it affects the results. This strengthens their understanding of how things work.

[3.DA.3c] Using computers to create models can be helpful because it allows us to work quickly and see results accurately, making it easier to understand complex systems. However, there are also drawbacks, such as relying on the quality of the input data and the risk of oversimplification, which can limit the model's ability to fully represent real-world scenarios.

Consider the following examples of some advantages and disadvantages to using computing devices to create models.

Advantages:

- Increased accuracy
- Faster
- Lower cost
- Can test design before building
- Identify potential problems before building

Disadvantages:

- Dependability on a computer
- Programming errors
- Data security
- Potential for data to be skewed

Concepts and Connections

CONCEPTS

Students organize and manipulate data to enhance understanding by utilizing various tools such as charts, graphs, and models. Models can be physical or digital to represent real-world objects or processes. Models help simplify complex systems and processes, providing a clearer understanding of how things work and how different elements interact within those systems.

CONNECTIONS

Within this grade level: At this grade level, students at this grade level create models of processes or physical objects and describe how computers are used to design the models. They also explore the advantages and limitations of using a computer to design models (3.DA.3).

Vertical progression: This is introduced at this grade level and is not addressed in Grade 3. Grade 4, students explore various models of physical objects or processes, create their own models, and use the models to explain how a computer model illustrates a concept (4.DA.3).

ACROSS CONTENT AREAS

Mathematics

- **3.NS.3** The student will use mathematical reasoning and justification to represent and compare fractions (proper and improper) and mixed numbers with denominators of 2, 3, 4, 5, 6, 8, and 10, including those in contextual problems.
- **3.CE.2** The student will recall with automaticity multiplication and division facts through 10×10 ; and represent, solve, and justify solutions to single-step contextual problems using multiplication and division with whole numbers.

Science

- 3.1 The student will demonstrate an understanding of scientific and engineering practices by e) developing and using models. *3.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 3.1 include: 3.2 (force, motion, energy), 3.5 (aquatic and terrestrial ecosystems), and 3.7 (water cycle).*

DIGITAL LEARNING INTEGRATION

- **3-5.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.
A. Formulate problem definitions suited for technology- assisted methods such as data analysis, modeling and algorithmic thinking in exploring and finding solutions.

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 construct mathematical models to represent fractions greater than one and analyze how changes in the numerator and/or denominator affect the model. Students then apply this understanding to predict and explain resulting changes in the fractional representations.

Science

- Students use computing devices to design a digital model of an animal habitat that incorporates a specific adaptation. They then analyze the model to make predictions about how the animal may respond or adapt to potential changes within the habitat.

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 Identify Patterns
3. Use 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

[Back to Data and Analysis \(DA\)](#)

Impacts of Computing (IC)

3.IC.1 The student will identify and examine the positive and negative impacts of the prevalence of computing technologies.

- a. Identify computing technologies that have changed the world.
- b. Examine and explain how computing technologies influence and are influenced by culture.
- c. Identify social and ethical issues related to the use of computing technologies.

Understanding the Standard

The development of computing technology has expanded exponentially over the past 100 years. This rapid growth has transformed the way people live, work, and connect with one another. This prevailing role of computing technologies and society has increased productivity, expanded communication, and improved access to information.

[3.IC.1a] Computing technologies have had a huge impact on our world. Key examples include the Internet, which connects people globally and provides access to information; smartphones, which make communication and information access easy and portable; and artificial intelligence, which helps with data analysis and automation. Other important technologies are cloud computing, which allows for flexible data storage; robotics, which improves efficiency in various industries; virtual reality, which creates immersive environments for education and training; and 3D printing, which allows us to create objects from digital designs.

[3.IC.1b] Computing technologies and culture influence each other. As technology evolves, it shapes how we communicate, learn, and interact, creating new social norms and behaviors. Cultural and societal needs inspire the development of new technologies that cater to changing demands and values. For example, social media has changed how we communicate and share information, affecting cultural norms around privacy and social interaction. At the same time, cultural trends drive the development of new technologies. For instance, the focus on sustainability has led to green computing innovations.

[3.IC.1c] The use of computing technologies brings up several important social and ethical concerns. Privacy is a major issue, as personal data can be misused or stolen. Cyberbullying and online harassment are significant problems, particularly for young people. The digital divide, which highlights unequal access to technology, can lead to inequalities in education and opportunities. Additionally, ethical dilemmas surrounding artificial intelligence and automation, such as job displacement and biased algorithms, require careful regulation to ensure fairness.

As computers continue to play a larger role in our daily lives, they have both positive and negative impacts. By understanding these effects, people can use technology in ways that maximize its benefits while minimizing its drawbacks. For example, to increase productivity,

individuals can use software to streamline tasks, while reducing negative effects by setting screen time limits to prevent eye strain and maintain a healthy balance between work and life.

Positive Impacts	Negative Impacts
• Quick Access to Information: We can find answers to our questions and learn new things easily. • Robots: Robots can help with repetitive or dangerous tasks, making jobs safer and healthier for workers. • Efficient Data Handling: Computers can process and analyze data quickly and accurately. • Helping People with Disabilities: Technology can make life easier for people with disabilities. • Screen readers: People with visual impairments can use computers and smartphones to access information. • Prosthetic limbs: Advanced prosthetics can help people with amputations regain mobility and independence. • Communication devices: People with speech impairments can use technology to communicate with others. • Global Communication: People can talk to friends and family anywhere in the world using video calls and messaging apps. • Access to Information: We can quickly find information on any topic using the Internet. • Automation: Machines can help with tasks, making work faster and more efficient.	• Distractions from Family Time: Technology can interrupt important moments with family and friends. • Privacy Concerns: Using computers can sometimes lead to personal information being shared without permission. • Screen Time and Health: Spending too much time on screens can have negative effects on physical and mental health. • Eye strain and headaches: Looking at screens for extended periods can cause eye strain, headaches, and blurry vision. • Sleep problems: The blue light emitted from screens can interfere with sleep patterns. • Lack of physical activity: Spending too much time sitting in front of screens can lead to a sedentary lifestyle and health problems. • Cyberbullying: Students can be harassed or bullied online through social media, text messages, or emails. • Inappropriate content: Students may come across inappropriate or harmful content online. • Privacy concerns: Students may share personal information online without understanding the potential risks.

• Learning New Things: Technology provides access to a vast amount of information and educational resources. • Educational apps and websites: Students can learn about different subjects in fun and interactive ways. • Online libraries and encyclopedias: Students can research topics for school projects or simply explore personal interests. • Videos and documentaries: Students can learn about history, science, and nature from experts around the world.	• Impact on the environment: The production and disposal of electronic devices can have negative environmental consequences. • Electronic waste: Nonworking computers, phones, and other devices often end up in landfills, where they can release harmful chemicals into the environment. • Mining for resources: The production of electronics requires the mining of rare earth minerals, which can damage ecosystems and pollute water sources. • Energy consumption: Using and charging electronic devices consume energy which can contribute to climate change. • Digital Divide: Not everyone has the same access to technology which can create inequalities. • Job Displacement: Automation and machines can sometimes replace jobs that people perform.
---	---

Concepts and Connections

CONCEPTS

Students will examine how computing technologies have transformed the world and society, analyzing their impact on and interaction with different cultures. They will also explore the social and ethical implications of these technologies, recognizing benefits and potential challenges.

CONNECTIONS

Within this grade level: At this grade level, students identify and examine the positive and negative impacts of the prevalence of computing technologies (3.IC.1).

Vertical progression: In Grade 2, students examine the positive and negative impacts of how using computing technologies has changed the way people live, work, and interact (2.IC.1). In Grade 4, students identify and examine the positive and negative impacts of the prevalence of computing technologies by identifying technologies that have impacted Virginia’s economy (4.IC.1).

ACROSS CONTENT AREAS

History and Social Science:

- **3.10** The student will apply history and social science skills to explain basic economic principles.
- **3.4** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Egypt.
- **3.5** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient China.
- **3.6** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Greece.
- **3.7** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of ancient Rome.
- **3.8** The student will apply history and social science skills to describe the geographic, political, economic, and social structures and innovations of the ancient empire of Mali.

DIGITAL LEARNING INTEGRATION

- **3-5.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.
 - A. Cultivate and manage their digital identity and reputation and are aware of the permanence of their actions in the digital world.
 - B. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.

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 explore the effects of technology and computing technologies on the production of goods and services as well as trading for specific cultures and societies.

Mathematics

- Students will conduct a survey or use provided data sets (e.g., screen time, energy use of devices, number of smart devices per household, time saved with online tools, etc.) to explore how computing technologies affect daily life. Students will organize the data into tables, charts, and graphs. They will use mathematical reasoning to compare values, calculate differences, and interpret trends (e.g., “How much more screen time do students spend per week compared to time outdoors?” or “What percentage of students say technology helps them learn better?”).

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

3.IC.2 The student will discuss and describe strategies to manage screen time.

- Define and describe screen time.
- Explain the importance of responsible screen time management.
- Discuss how screen time choices affect one's personal health and interactions with others.

Understanding the Standard

[3.IC.2a] Screen time refers to the amount of time spent using digital devices such as computers, tablets, and smartphones for various activities, including playing games, watching videos, browsing the internet, and engaging in educational tasks. It is important to monitor and manage screen time to ensure a balanced lifestyle and promote healthy habits.

[3.IC.2b] Managing screen time responsibly is crucial for health and well-being, helping to prevent issues like eye strain, reduced physical activity, and disrupted sleep patterns. Setting boundaries on screen use, such as taking regular breaks, engaging in physical activities, and ensuring adequate sleep, can promote a balanced lifestyle and support overall cognitive development.

To manage screen time effectively, students can set limits using a timer, take breaks, create screen-free zones, and balance screen time with physical activity. By incorporating these strategies, students can develop healthy habits and use technology responsibly.

Strategy	Example
Setting Limits	Use a timer to limit screen time to 30 minutes per session.
Taking Breaks	Follow the 20-20-20 rule: every 20 minutes, look at something 20 feet away for 20 seconds.
Choosing Educational Content	Select educational apps and websites, like a math game app for practicing addition.
Creating a Screen-Free Zone	Designate areas like the dining room or bedrooms as screen-free zones.
Balancing Screen Time with Activity	After 30 minutes of screen time, spend 30 minutes playing outside or riding a bike.

[3.IC.2c] Screen time choices can have both positive and negatives impact on personal health and social interactions. On the positive side, screens can enhance social interactions, facilitate global connectivity and foster online collaboration. Excessive screen use can have negative effects leading to physical and mental health concerns, as well as impacting peer relationships.

Concepts and Connections

CONCEPTS

Students will evaluate strategies for managing screen time, including defining the concept and understanding its importance. They will analyze how responsible screen time management contributes to personal well-being and examine its effects on social interactions and health.

CONNECTIONS

Within this grade level: At this grade level, students discuss and describe strategies to manage screen time by explaining the importance of responsible screen time management as well as discuss how screen time impacts one's health (3.IC.2).

Vertical progression: In Grade 2, students explain the need to balance screen time with other activities (2.IC.2). In Grade 4, Students describe the impact of screen time on relationships at home and at school (4.IC.2).

ACROSS CONTENT AREAS

English

- **3.C** The student will develop effective oral communication and collaboration skills to build a community of learners that process, understand, and interpret content together.

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 conduct a poll to collect data on how much time their classmates spend using different types of technology and how their screen choices relate to their daily activities. Using this data, students create a bar graph or pictograph to visually represent screen time habits and explore the relationship between screen time management and personal well-being.

History and Social Science

- Students identify reasons sleep is important and the consequences of excessive use of computing devices. Students create individual goals to improve health and track their progress with a weekly checklist.

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.IC.3 The student will identify and describe computing technology careers and their impact on society.

- Research computing technology careers.
- Describe the impact careers in computing technology have on society.

Understanding the Standard

[3.IC.3a] Careers in computing technology are essential in shaping the future of society by addressing societal needs, improving quality of life, and opening up new opportunities for education, business, and communication. For instance, medical software helps doctors figure out what is wrong with patients, educational apps make learning fun for students, and video game designers create games that are both entertaining and educational.

When researching careers like software engineers, UX/UI designers, cybersecurity analysts, and video game designers, students should start by using reliable sources such as government websites, industry associations, and educational institutions to gather accurate information. Students should research the required skills, education, and certifications for each role, as well as potential career growth opportunities.

[3.IC.3b] Careers in technology have a significant impact on society in various ways. They drive innovation and progress, creating new tools and solutions that improve daily life, enhance productivity, and solve complex problems.

Career	Impact on Society	Real-World Examples
Software Engineer	Develop apps that help people stay connected and productive.	Apps for video calls, messaging, and productivity tools
Web Developer	Builds online tools and platforms for collaboration and information sharing.	Online document editors and project management tools
Video Game Designer	Create games that entertain and educate, enhancing learning and creativity.	Educational games for reading, math, science, and history
Medical Software Developer	Designs software that assists doctors in diagnosing and treating patients.	Medical imaging software and electronic health records
Educational App Developer	Develop apps that make learning interactive and fun for students.	Language learning apps and math practice apps
Cybersecurity Experts	Create or monitor systems used to protect data from unauthorized users and data breaches.	Cybersecurity analysts protect an organization's digital infrastructure by

		monitoring networks, identifying threats, and responding to cyberattacks.
--	--	---

Concepts and Connections

CONCEPTS

Understanding careers in computing technology is essential as these fields play a pivotal role in shaping advancements across various industries and sectors. Knowledge of these careers provides students the opportunity to understand the impact of technology on society and explore potential career pathways.

CONNECTIONS

Within the grade level/course: At this grade level, students identify and describe computing technology careers and their impact on society (3.IC.3).

Vertical Progression: In Grade 2, students explain how computing technologies have an impact on the workforce (2.IC.3). In Grade 4, Students examine the impact of computing technologies in the workforce (4.IC.3).

ACROSS CONTENT AREAS

English

- **3.C.1A** Participate in a range of collaborative discussions (one-on-one, in groups, and teacher-led) on grade three topics and texts.
 - iii. Asking and responding to questions that acquire or confirm information a topic and link their comments to the remarks of others.

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 describe how ancient people adapted to their environments, focusing on technological advancements, Greek and Roman innovations and their influences.

Science

- Students can explore how simple machines reduce force and make work easier. They will examine how these machines have evolved into more complex tools and how careers like construction, farming, and medicine have changed as a result. Students will also investigate how computing technologies such as coding, robotics, and sensors are now used to control or improve these machines. By comparing past and present tools, students will understand how science and computing shape the way people work.

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

3.IC.4 The student will demonstrate how to use information created by others with permission.

- a. Discuss copyright, piracy, and plagiarism.
- b. Demonstrate how to use information created by others.

Understanding the Standard

The creation of original work involves generating unique ideas, content, or products that reflect an individual's or group's intellectual effort. It is essential to recognize and respect the ownership of such work by properly citing sources and giving credit to original creators. This practice ensures intellectual property rights are upheld.

[3.IC.4a] Understanding the concepts of copyright, piracy, and plagiarism is essential. Copyright protects the rights of creators by giving them control over how their work is used. Piracy involves the unauthorized use or reproduction of someone else's work, while plagiarism is the act of presenting someone else's work or ideas as one's own without giving proper attribution.

[3.IC.4b] There are guidelines that students must follow to properly use information created by others.

Provide Proper Attribution: Always credit the original creators through citations, references, or attributions.

- **Seek Permission:** When necessary, obtain permission from the copyright holder to use their work. This may involve direct communication with the creator or entering into licensing agreements.
- **Understand Fair Use:** Familiarize oneself with the concept of fair use, which permits limited use of copyrighted material without permission under specific conditions, such as for educational purposes, commentary, or criticism.

Concept	Explanation	Example
Copyright	Legal protection for creators, giving them exclusive rights to their work.	A student uses an image from a website in a school project and credits the photographer.
Piracy	Unauthorized use or reproduction of someone else's work.	Downloading a movie without paying for it or without the creator's permission.
Plagiarism	Presenting someone else's work or ideas as your own without giving credit.	Copying text from a book or online resource into a report without citing the source.

Giving Credit	Acknowledging the original creator of the work you are using.	Including a bibliography in a research project that lists all the sources used.
Obtaining Permission	Asking for and receiving permission to use someone else's work.	Email an author to ask if you can use their article in your class presentation.

Concepts and Connections

CONCEPTS

It is essential to adhere to copyright laws when using others' work, ensuring that intellectual property is respected. Students should credit original creators, obtain necessary permissions, and use content that is publicly available or licensed for reuse.

CONNECTIONS

Within the grade level/course: At this grade level, students demonstrate how to use information created by others with permission, and discuss copyright, piracy, and plagiarism (3.IC.4).

Vertical Progression: This is introduced at this grade level and is not addressed in Grade 3. In Grade 4, students describe the importance of copyrights and intellectual property rights, demonstrate an understanding of copyright and the fair use of information, explain how intellectual property can be protected, and give proper attribution to the original author of digital and online content (4.IC.4).

ACROSS CONTENT AREAS

English

- **3.C.1A** Participate in a range of collaborative discussions (one-on-one, in groups, and teacher-led) on grade three topics and texts.
 - iii. Asking and responding to questions that confirm information on a topic and linking their comments to the remarks of others.

DIGITAL LEARNING INTEGRATION

- **3-5.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. Demonstrate an understanding of and respect for the rights and obligations of using and sharing intellectual property.
- **3-5.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.

- B. Evaluate the accuracy, perspective, credibility, and relevance of information, media, data, and other digital sources.
- **3-5.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. Create original works or responsibly repurpose or remix digital resources into new creations.

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 social and ethical issues related to online communication (e.g., cyberbullying, misinformation, privacy) using credible sources such as articles, videos, or expert interviews. In small groups or individually, students will identify a key issue, cite at least one source to support their understanding, and propose a responsible solution.

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

[Back to Impacts of Computing \(IC\)](#)

Networks and the Internet (NI)

3.NI.1 The student will describe computing networks.

- a. Differentiate between a network and the Internet.
- b. Identify the components of a computing network.
- c. Describe how a computing device connects to a network.
- d. Identify ways networks are used to transmit information.

Understanding the Standard

The Internet is the underlying infrastructure that connects devices, allowing them to communicate with one another. A network is a system of interconnected devices that can share resources and exchange information within a specific area, like a local area network (LAN) or even the global network that forms the Internet. The Internet gives people access to the web, a system of interconnected resources where information can be found on various platforms. The World Wide Web acts as a digital library, providing access to a wide range of data and information.

- The Internet is a global system of interconnected networks that allows data to travel across vast distances, enabling communication and information exchange. A network, on the other hand, is a smaller, localized system of connected devices that allows them to share resources and communicate with each other. While the Internet is a network of networks, a network refers to a specific group of connected devices or systems within a more limited scope, like a local area network (LAN) in a home or office. The Internet uses these networks to facilitate global connectivity and the transfer of information.

[3.NI.1a] A network is a system of interconnected devices that communicate with each other, either through physical connections like wires or through wireless signals. Networks can be small, like the one in a school or home, or large, like the one that connects a city or a country. The Internet, on the other hand, is a global network of networks that connects millions of devices worldwide. It serves as an infrastructure that allows data and information to be exchanged across different networks, enabling people to access websites, send emails, and use online services. While networks operate locally or regionally, the Internet provides the overarching connection between these smaller networks, allowing communication on a global scale.

Differentiate between a network and the Internet:

- Network: A network is a group of connected devices that can communicate with each other, like computers, printers, and servers within a school or office.
- Your home Wi-Fi is a network made up of your phone, laptop, and smart TV which can talk to each other.

- Internet: The Internet is a vast global network of interconnected networks, allowing devices worldwide to communicate. It's like a giant web that connects millions of smaller networks together.
- When you browse a website, you're using the Internet your device connects to a distant server outside your home network.

[3.NI.1b] A network typically consists of several key computing devices, such as routers, switches, and servers that enable communication between different systems

- Router: Think of a router as a traffic director that helps data find its way to the right device. At home, your router connects all your family's devices to the Internet.
- Switch: A switch is like a power strip for data, allowing multiple devices to connect to the network at the same time. In your school, a switch connects several computers to the network.
- Server: A server is like a library that stores and provides information. When you access a website, you are getting information from a server.

[3.NI.1c] Computing devices are connected through physical cables or wirelessly through technologies enable devices to send and receive data across the network, ensuring seamless communication and access to resources.

- Physical Cables: Connecting a device with an Ethernet cable is like plugging in a lamp to get electricity. For example, your gaming console might use an Ethernet cable to connect to the Internet for online play.
- Wireless Signals: Connecting a device via Wi-Fi is like using a walkie-talkie to communicate without wires. Your tablet connects to the Internet wirelessly at home or in a café.

[3.NI.1d] Networks transmit information through many methods. Wi-Fi connects devices like smartphones and laptops to the internet, while Bluetooth allows short-range communication, such as linking wireless headphones to a tablet. Cellular networks help mobile phones send calls and texts by connecting to nearby towers. In addition to these everyday examples, fiber optic cables laid deep under the ocean carry internet data between continents allowing for global connectivity.

Concepts and Connections

CONCEPTS

A network is a system that connects computing devices, allowing them to share and exchange information. When thousands of individual networks, including local, regional, and global systems, are linked together, they form the Internet. Networks are of routers, servers, cables, and wireless connections and provide the essential structure that enables the transmission of data/information.

CONNECTIONS

Within the grade level/course: At this grade level, students describe computing networks, differentiate between a network and the Internet, identify the components of a computing network, describe how a computing device connects to a network, and identify ways networks are used to transmit information (3.NI.1).

Vertical Progression: In Grade 2, students demonstrate the use of the Internet in gathering information to accomplish a task (2.NI.1). In Grade 4, students identify the interrelationship between computing devices and a computing network; define the roles of client and server; describe how packets are used to transmit information on a network; describe factors that may affect the speed of data transmission; and differentiate between networking tasks that require Internet access and tasks that do not require Internet access (4.NI.1).

ACROSS CONTENT AREAS

English:

- **3.C.1A** Participate in a range of collaborative discussions (one-on-one, in groups, and teacher-led) on grade three topics and texts.
 - iii. Asking and responding to questions that acquire or confirm information on a topic and link their comments to the remarks of others.

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 will simulate how data travels through a computer network by participating in a timed relay, transferring "data packets" along a marked path. They will record how many packets were moved and how long it took, then calculate the rate of transfer (e.g., packets per minute). Using multiplication and division, students will solve related math problems and compare team results using charts or graphs. Students can also reflect on what made some teams more efficient, connecting their observations to how real computer networks handle data traffic.

Science

- Students explore how data flows through a network by comparing it to an ecosystem. They will learn how devices, like computers and routers, are similar to producers and animals that work together to send and receive information, just like how species in an ecosystem depend on each other.

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 Identify Patterns
3. Use Abstraction to Simplify, Represent, and Problem Solve

[Back to Networks and the Internet \(NI\)](#)

Appendix A

K-5 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 (e.g. visualizations, storyboards, flowcharts, decision trees, models, computer simulations).

A. Fostering Collaboration in Computing Practices

1. Building Relationships and Norms:

- **K-2:** Students work collaboratively with others. Students take turns in different roles on the project.
- **3-5:** Students work collaboratively with others. Students practice assigning roles within their teams and recognize group member strengths.

2. Include Multiple Perspectives:

- **K-2:** Students differentiate their technology preferences from the technology preferences of others. Students will be presented with perspectives from people with different backgrounds, ability levels, and points of view.
- **3-5:** Students discuss design choices, compare preferences, ask questions, and seek input from group members with diverse abilities, experiences, and perspectives.

3. Create and Accept Feedback:

- **K-2:** With teacher scaffolding, students seek help and share ideas to achieve a particular purpose. Students ask questions of others and listen to their opinions.
- **3-5:** Students provide and receive feedback related to computing in constructive ways. For example, pair programming is a collaborative process that promotes giving and receiving feedback.

4. Use Collaboration Tools:

- **K-2:** Students collaboratively brainstorm by writing on a whiteboard or paper.
- **3-5:** Students use collaboration tools to manage teamwork and utilize online project spaces. They also begin to make decisions about which tools would be best to use and when to use them.

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 different 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 Discussion:** Organize discussions that engage students in hearing differing 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 Real-World Problems:

- **K-2:** Students break problems, information, and processes into parts. Identify relationships and connections among parts. Reflect on how decomposition aids problem-solving across contexts.
- **3-5:** Students further break problems into subproblems, apply systems thinking to explore interdisciplinary connections and integrate existing solutions or procedures (i.e. classroom processes, math procedures, school routines) Apply algorithms to break a problem into subtasks that can be solved and combined to solve the main problem.

2. Explore Common Features and Identify Patterns:

- **K-2:** Students will be able to identify and describe repeated sequences in data or code through analogy to visual patterns or physical sequences of objects. Students will identify patterns, such as recognizing repeated patterns of code that could be more efficiently implemented as a loop.
- **3-5:** Students analyze patterns to develop generalizations and models, test their limits, and validate inputs. Use patterns to analyze trends, justify design decisions, and create artifacts.

3. Use Abstraction to Simplify, Represent, and Problem Solve:

- **K-2:** Students use and/or create abstractions (e.g. storyboards, flowcharts, decision trees, models) to simplify problems, represent information, organize thinking, communicate, and create artifacts. Artifacts can include but are not limited to prototypes, programs, planning documents, and animations.
- **3-5:** Students use and/or create abstractions (e.g. visualizations and computer simulations) to simplify problems, represent information, organize thinking, communicate, and create artifacts. Students intentionally use abstractions to support the problem-solving process to aid in understanding, planning, and predictions.

4. Apply Algorithmic Thinking to Problem Solve and Create:

- **K-2:** Students use algorithmic thinking to develop a sequence of steps to plan, create, test, and refine artifacts with and without technology.
- **3-5:** Students use pseudocode and generalizations to organize, create and seek and incorporate feedback on more complex designs.

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

- **K-2:** Students use computational thinking to organize data and make predictions. Explore parts and relationships within data sets.
- **3-5:** Students visualize data. 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.

Instructional Considerations for Computational Thinking 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 an Artifact:** 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 choose the best tool to use for problem solving using abstractions. Discuss which tools worked best for the team and the problem. Tools may include models, visualizations, storyboards, flowcharts, decision trees, generalizations, simulations.
- **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:** 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 apply the practice to other situations. Support students identify key points in feedback.

C. Fostering Iterative Design Practices

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

- **K-2:** With guidance from an educator, identify, define, and explore existing problems and potential solutions. Ask questions to view problems from different perspectives.
- **3-5:** Students identify, define, and explore existing problems and potential solutions. Ask questions to understand 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.

2. Plan and Design Artifacts:

- **K-2:** With guidance from an educator, students will generate ideas for new solutions, incorporate peer feedback, and reflect on impact of diverse perspectives. Use tools like class or group discussions, outlines, flowcharts, and storyboards to plan prototypes.
- **3-5:** Students will generate ideas for new solutions, incorporate peer feedback, and reflect on the impact of diverse perspectives. Use tools like outlines, flowcharts, and storyboards to plan prototypes. Predict the performance and impacts of prototypes, including potential errors, user needs, and accessibility.

3. Create, Communicate and Document Solutions:

- **K-2:** Students create artifacts with or without technology, such as algorithms and programs using plans and outlines. Describe design choices and make connections to the design challenge, criteria, and constraints. Engage in giving and receiving feedback enhances communication skills.
- **3-5:** Students create artifacts, such as algorithms and programs using plans and outlines. 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.

4. **Test and Optimize Artifacts:**

- **K-2:** 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.
- **3-5:** 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.

Instructional Considerations for Iterative Design 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 an 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 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 Discussions:** Discuss 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, and utilize coding platforms to simulate solutions. Incorporate available technology to develop physical models. Use peer feedback to refine designs, and document changes while justifying improvements at each step.

- **Debug and Enhance:** Work with a pre-built program containing intentional errors and limited features to debug to 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 iterative changes to improve the design, such as adding features like text-to-speech, adjustable font sizes, or simplified navigation and implement when available and appropriate.

D. Fostering Digital Literacy Practices

1. Responsible Use Practices:

- **K-2:** Students use technology in ways that are safe, legal, and ethical. Implement strategies to protect their digital identity, personal data, and the data of others.
- **3-5:** Explore and ask questions about how computer science and emerging technologies work, and their benefits and risks. Students explore data privacy rights, data protections, terms of service and privacy policies. Weigh tradeoffs and risks with actions and decisions involving computer science.

2. Safeguard Well-Being of Self and Others:

- **K-2:** Students reflect on their emotional response to the use of digital technology. 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. Practice empathy and engage in positive online practices as an upstander.
- **3-5:** 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. Practice empathy and engage in positive online practices as an upstander.

3. Evaluate Resources and Recognize Contributions:

- **K-2:** Students apply strategies for evaluating the accuracy, validity, accessibility, reliability, appropriateness, credibility, and relevance of digital sources.
- **3-5:** Students apply strategies for evaluating the accuracy, validity, accessibility, reliability, appropriateness, credibility, and relevance of digital sources. Keep track of sources of information and give credit to the creators of information. Students evaluate the bias and relevance of sources. Identify false or misleading information.

Instructional Considerations for Digital Literacy 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 identity 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.

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.
- **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 Discussions:** Organize discussions 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 3 Computer Science Vocabulary

Vocabulary Word	Definition
Abstraction	A filtering process used to create a simplified representation of relevant data to identify essential details, excluding less important details.
Algorithm	Finite and specified set of step-by-step instructions designed to solve a problem or perform a task.
Algorithmic Thinking	Process of developing algorithms in a logical, systematic, and procedural way to solve problems or complete tasks.
Acceptable Use Policy (AUP)	Rules and guidelines that define safe practices and responsible use of technology.
Attribution	Giving credit for work created by someone else.
Authentication	A process used to verify a user's identity before accessing a network or computer system.
Author	The creator of a book, image, song, or object.
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.
Categorical Data	Are observations about characteristics that can be sorted into groups or categories (e.g., qualitative). It may include photos, text images, videos, non-numerical values, survey responses, true/false values, or colors.
Character	A person or animal in a book, story, movie, or project. A letter, number, or symbol used in a password.
Code	Any set of instructions expressed in a programming language.
Collecting	Gathering the appropriate type of data needed.
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 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	An electronic device that can receive input, process data, store information, and produce output based on instructions (programs).
Conditional Control Structures	Conditional logic (e.g., if-else statements) to make decisions within a computer program.
Copyright	Legal protection that gives creators of original works (like literature, art, music, software, etc.) exclusive rights to use and distribute their creations.
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 Cycle	Process of formulating questions to be explored with data, collecting or acquiring data, organizing and representing data, and analyzing and communicating results.
Data Representation	How data is visually represented, such as in graphs or charts.
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.
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.
Decomposition	Process of breaking down a problem, process, or task into smaller, more manageable components.
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.
Diagrams	Visual representation of data, information, or concepts.
Email	A program used to send and receive messages over the Internet for online communication.
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.
Event	An action or something that causes all of a program or only a certain portion of the program to run, e.g., a “mouse click” on the run block.
Fair Use	Allows the limited use of copyrighted material without the copyright owner's permission.
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.
Healthy Screen Habits	Practices that emphasize balanced use of digital devices to support physical, mental, and emotional well-being.
Implementation	The development or execution of a functional prototype, program, or product.
Information	Facts provided or learned about something or someone.
Input	Information or action you give to a computer or device to tell it what to do.
Input Devices	Hardware components that allow users to enter data into a computing device.
Intellectual Property	A person's own creations of the mind, such as inventions, drawings, stories, and poems.
Internet	A global network of interconnected computing devices that allows devices to share information and resources.
Iteration	Repeated actions.
Iterative Design Process	A systematic approach to creating and refining products, systems, or solutions through repeated cycles of design, evaluation, and improvement.
Key	A distinct identifier used to differentiate data elements within a set.
Library	A collection of books and periodicals.

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.
Loop	A set of instructions that are repeated until a specified condition is met, or a predetermined number of repetitions has occurred.
Memory	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.
Model	A simplified representation of an idea, object, system, or process that helps describe, test, or predict how something works often using diagrams, simulations, or code.
Networking	A group of computing devices (personal computers, phones, servers, switches, routers, etc.) connected by cables or wireless media for the exchange of information and resources.
Numerical Data	Are values or observations that can be measured (e.g., quantitative). It may include heights, temperatures, scores or grades, or statistics.
Output	Data or information produced by a computing device after processing input.
Output Devices	Hardware components that display processed data or information.
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.
Personal Information	Data or information about a person that relates to their identity, characteristics, or activities.
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.
Private Information	Sensitive data or information that can identify a person or give others access to your personal life.
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.

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	A structured system for writing instructions that a computer can understand and execute. It includes syntax, which defines the rules for how code is written, and semantics, which conveys the meaning of the instructions. Programming languages enable developers to build software, automate processes, and control computer hardware.
Prototype	A preliminary version of a final product or information system, typically created for demonstration purposes.
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.
Reboot	To turn off the device and turn it back on.
Robotics	Using robots to perform repetitive tasks with precision.
Router	A device that directs data from one network to another.
Screen Time	Time spent on a computing device.
Selection	Using conditions to manage the sequence of a program's execution.
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.
Simulation	Replicating the behavior of a real-world process or system over a period of time.
Social Media	Applications that allow people to socialize, communicate, and share content with each other.
Software	A set of instructions that tells the computer how to act and respond but cannot be seen or touched.
Sort	To compare a set of objects in order to find similarities and differences, so that they may be arranged and organized.
Storage	Location where data, programs, and files are kept permanently (until deleted).
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.

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.
Texting/text Message	A short, written message sent from one phone to another over the Internet or a mobile network.
Trends	Are long-term directions or movements in data or behavior that indicate a general tendency or shift over time.
Troubleshoot	Processes 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.
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.
Video Call	A live conversation where people can see and hear another person while talking through a connected device, like a tablet or computer.
Video Conference	A meeting where people in different locations use video and audio technology to communicate in real-time allowing participants to see and hear each other as if they were in the same room.
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.
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.
World Wide Web (WWW)	A system of interconnected web pages that users can access through the internet.