



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 of Contents

2024 Computer Science Standards of Learning	3
Introduction	3
Foundational Principles	3
Fourth Grade: 2024 Computer Science Standards of Learning	5
Algorithms and Programming (AP)	5
Computing Systems (CSY)	6
Cybersecurity (CYB)	6
Data and Analysis (DA)	7
Impacts of Computing (IC)	8
Networks and the Internet (NI)	8
Computer Science Instructional Guide Framework	10
Grade 4: Computer Science Instructional Guide	11
Algorithms and Programming (AP)	11
Computing Systems (CSY)	31
Cybersecurity (CYB)	45
Data and Analysis (DA)	56
Impacts of Computing (IC)	73
Networks and the Internet (NI)	87
Appendix A	91
K-5 Computer Science Skills and Practices Continuum	91
Appendix B	99
Grade 4 Computer Science Vocabulary	99

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.

Fourth Grade: 2024 Computer Science *Standards of Learning*

In Fourth Grade, students deepen their comprehension of the intricate connection between computing devices and networks. The exploration of cybersecurity becomes a focal point, guiding students in understanding the responsible use of computing technologies and the repercussions of inappropriate usage. The students investigate the mechanisms through which computers perceive the world, incorporating sensors and various inputs into their understanding. Computational practices progress as algorithms includes variables and user input. The utilization of data is broadened, with a focus on identifying appropriate data types for problem-solving and creating models of real-world situations.

Algorithms and Programming (AP)

4.AP.1 The student will apply computational thinking to identify patterns and design algorithms to compare and contrast multiple algorithms used for the same task.

- a. Decompose an algorithm, process, or problem into a subset of smaller problems.
- b. Identify multiple algorithms for the same task.
- c. Describe patterns within multiple algorithms.
- d. Determine which algorithm is most effective for a given task.

4.AP.2 The student will plan and implement algorithms that consist of sequencing, loops, variables, user input, and conditional control structures using a block-based programming language.

- a. Identify user input and its role in improving a program.
- b. Describe the concept of a variable.
- c. Read and explain a design document to trace and predict an algorithm using plain language, pseudocode, or diagrams.
- d. Create a design document to plan an algorithm using plain language, pseudocode, or diagrams.
- e. Write programs that initialize, assign values to, name, and modify variables.

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

- a. Create and test programs that consist of sequencing, loops, variables, user inputs, and conditional control structures.
- b. Create and use variables to store and process data.
- c. Trace and predict the value of variables that change over the course of the program's runtime.

- d. Analyze and describe program results to assess validity of outcomes.
- e. Revise and improve programs to resolve errors or produce desired outcomes.

Computing Systems (CSY)

4.CSY.1 The student will model how a computing system works to accomplish a task.

- a. Describe how computing systems perceive the world through sensors and other inputs.
- b. Compare and contrast how humans and computers process information from inputs.
- c. Explain how computing devices may be used to classify and organize input.
- d. Diagram and describe a simple computing system indicating processors, inputs, and outputs.

4.CSY.2 The student will apply troubleshooting strategies when a computing system is not working as intended.

- a. Identify hardware, software, and connectivity problems using accurate terminology.
- b. Apply troubleshooting strategies to address hardware, software, and connectivity problems.

4.CSY.3 The student will describe the learning process of humans and computers.

- a. Compare and contrast how humans and computing technologies collect, store, and process data.
- b. Identify similarities and differences on how humans and computing technologies infer and extract meaning from data.
- c. Define machine learning and identify machine learning approaches: supervised, unsupervised, and reinforcement learning.

Cybersecurity (CYB)

4.CYB.1 The student will examine the impacts of appropriate and inappropriate use of computing technologies.

- a. Examine and explain scenarios for appropriate and inappropriate use of computing technologies.
- b. Develop possible solutions involving inappropriate use of computing technologies.

4.CYB.2 The student will identify and investigate best practices to safeguard information shared online and through online platforms.

- a. Classify personal, private, and public information.
- b. Research and evaluate tradeoffs of sharing information.
- c. Investigate and communicate best practices to limit unauthorized access to information on a computing device.
- d. Demonstrate proper use and protection of personal passwords.
- e. List methods used to safeguard online information.

4.CYB.3 The student will examine how information is shared online and explain the importance of cybersecurity.

- a. Investigate multiple ways people share information online.
- b. Determine and describe when information should be shared and to whom it should be shared.
- c. Describe how personal information can be collected and shared online.
- d. Explain the importance of cybersecurity.

Data and Analysis (DA)

4.DA.1 The student will identify the appropriate type of data needed to solve a problem or answer a question.

- a. Analyze a problem to determine the appropriate type of data needed.
- b. Evaluate the reliability of data sources.
- c. Use numeric values to represent non-numeric ideas to include binary, American Standard Code for Information Interchange (ASCII), and RGB values.
- d. Collect, store, clean, and organize data for analysis and to prepare visualizations.

4.DA.2 The student will create and evaluate data representations to make predictions and conclusions.

- a. Formulate questions that require the collection or acquisition of data.
- b. Collect data to create charts and graphs.
- c. Recognize and analyze patterns and relationships within data sets.
- d. Analyze visual representations to make predictions and draw conclusions.

4.DA.3 The student will create a computational model that represents attributes and behaviors associated with a concept.

- a. Examine models of physical objects and processes.
- b. Create a computational model that reflects the attributes and behaviors associated with a concept.
- c. Explain how a computer model illustrates a given concept.

Impacts of Computing (IC)

4.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 Virginia's economy.
- b. Examine and explain how computing technologies influence and are influenced by culture.
- c. Identify social and ethical issues related to computing devices and networks.

4.IC.2 The student will describe the impact of screen time on relationships at home and at school.

- a. Describe the impact of excessive screen time on maintaining friendships and family dynamics.
- b. Explain how playing video games and the use of social media can impact relationships and personal health.

4.IC.3 The student will examine the impact of computing technologies in the workforce.

- a. Research and analyze the skills needed for careers in computing technology fields.

4.IC.4 The student will describe the importance of copyrights and intellectual property rights.

- a. Demonstrate an understanding of copyright and the fair use of information.
- b. Explain how intellectual property can be protected.
- c. Give proper attribution to the original author of digital and online content.

Networks and the Internet (NI)

4.NI.1 The student will identify the interrelationship between computing devices and a computing network.

- a. Define client and server.
- b. Describe how packets are used to transmit information on a network.
- c. Describe factors that may affect the speed of data transmission.
- d. Differentiate between networking tasks that require Internet access and tasks that do not require Internet access.
- e. Model how computing devices in a network transmit and receive 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 4: Computer Science Instructional Guide

In Grade 4 students deepen their comprehension of the intricate connection between computing devices and networks. The exploration of cybersecurity becomes a focal point, guiding students in understanding the responsible use of computing technologies and the repercussions of inappropriate usage. The students investigate the mechanisms through which computers perceive the world, incorporating sensors and various inputs into their understanding. Computational practices progress as algorithms includes variables and user input. The utilization of data is broadened, with a focus on identifying appropriate data types for problem-solving and creating models of real-world situations.

Algorithms and Programming (AP)

4.AP.1 The student will apply computational thinking to identify patterns and design algorithms to compare and contrast multiple algorithms used for the same task.

- a. Decompose an algorithm, process, or problem into a subset of smaller problems.
- b. Identify multiple algorithms for the same task.
- c. Describe patterns within multiple algorithms.
- d. Determine which algorithm is most effective for a given task.

Understanding the Standard

Computational thinking (CT) is a logical and systematic problem-solving process that uses decomposition, pattern recognition, abstraction, and algorithm 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.

The four main computational thinking components are 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 creation of step-by-step instructions or algorithms to solve the problem or task.

[4.AP.1a] The process of breaking down complex problems or tasks into smaller, more manageable components is known as decomposition. Decomposition allows students to create organized and effective algorithms. The process of decomposition is essential in algorithmic design as it helps students identify and organize logical sequences that are required to solve larger or multi-step problems or tasks. This practice supports the development of systematic thinking by encouraging students to focus on individual components before considering the problem as a whole. It lays the foundation for debugging and optimization, as students can more easily identify errors or inefficiencies within smaller parts of an algorithm.

[4.AP.1b] Many tasks or problems can be solved in multiple ways, each providing a different path to the same outcome. This versatility allows for the implementation of various algorithms to accomplish the same task. Once a task is broken into subsets of smaller problems, students recognize that one task can be completed using multiple algorithms.

Many tasks or problems can be solved in multiple ways, each providing a different path to the same outcome. This versatility allows for the implementation of various algorithms to accomplish the same task. Once a task is broken into subsets of smaller problems, students recognize that one task can be completed using multiple algorithms. Identifying multiple algorithms involves analyzing the task requirements, considering different logical sequences of steps, and exploring alternative approaches that achieve the same goal. Students may compare algorithms based on criteria such as efficiency, clarity, number of steps, or resource use (e.g., time or memory), which fosters critical evaluation skills. Through this process, students build a deeper understanding of computational thinking by examining trade-offs between different solutions and justifying their algorithmic choices based on context.

Abstraction is used when students examine and compare different algorithms, enabling them to concentrate on the fundamental ideas that make each solution work while setting aside unnecessary details. By recognizing shared patterns, structures, and underlying logic across multiple approaches, students gain insight into the core principles that guide algorithmic design. This analytical process strengthens their ability to generalize solutions and promotes higher-order thinking. As a result, students refine their problem-solving strategies and deepen their computational thinking, equipping them to identify efficient, adaptable solutions across a variety of contexts.

[4.AP.1c] Describing patterns within multiple algorithms involves analyzing their structural similarities, logical flow, and the strategies they use to solve a given problem. Students can begin by identifying common elements such as the sequence of steps, use of loops or conditionals, and the way inputs are processed to produce outputs. By comparing these elements across different algorithms, students are able to articulate how certain problem-solving techniques recur, even when surface-level details vary. This practice requires attention to both the procedural components and the conceptual strategies embedded within each algorithm. Recognizing and describing these patterns not only deepens students' understanding of algorithmic design but also promotes abstraction and reinforces their ability to generalize knowledge to new computational problems.

[4.AP.1d] Determining the most effective algorithm for a task means understanding what the task requires, how different algorithms work, and why one might be better than another. Students start by thinking about what the algorithm needs to do and what information it uses. This helps them think critically, solve problems more effectively, and understand how to improve their work. A clear understanding of what information or data is most necessary in an algorithm enables students to determine which algorithm is most effective for a given task. By way of example, a student may determine that one algorithm is too repetitive, while another algorithm uses loops to perform the same task more efficiently. By learning to determine which algorithm is most effective, students gain the tools they need to make better decisions, optimize their solutions, and think critically.

For example: Students can explore two different methods to sort a group of numbers (4, 2, 3, 8, 10).

1. Step-by-step comparison method: Looking at each pair of numbers one by one and swapping them when needed until the entire set is in order.
2. Grouping and sorting method: Splitting numbers into two smaller groups, sorting each separately, then combining them in order.

Comparison of these two methods, such as which is faster, which requires more steps, which is easier to follow, enables students the ability to see how different algorithms lead to the same correct outcome, while offering varying levels of efficiency and complexity.

Concepts and Connections

CONCEPTS

Understanding algorithms involves decomposition and exploration of multiple solutions for the same task. Recognizing underlying patterns across different approaches allows for an evaluation of efficiency, accuracy, and logical consistency. This process enhances computational thinking, enabling the application of systematic problem-solving techniques across a range of contexts.

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 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).

Vertical Progression: In Grade 3, students used pattern analysis, decomposition, and algorithmic thinking to design algorithms that extend patterns, processes, or components of a problem (3.AP.1). In Grade 4, students will apply abstraction as they design an algorithm to solve a problem (5.AP.1).

ACROSS CONTENT AREAS

English

- **4.RV** The student will systematically build vocabulary and word knowledge based on grade-four content and texts.

(Note: students are using content vocabulary when describing patterns/algorithms/etc.)

Mathematics

- **4.NS.1c** The student will use place value understanding to read, write, and identify the place and value of each digit in nine-digit whole numbers.
- **4.CE.4** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using addition and subtraction of decimals through the thousandths, with and without models.
- **4.PFA.1** The student will identify, describe, extend, and create increasing and decreasing patterns (limited to addition, subtraction, and multiplication of whole numbers), including those in context, using various representations.
- **4.NS.3f** The student will use mathematical reasoning and justification to represent, compare, and order fractions (proper, improper, and mixed numbers with denominators 12 or less), with and without models.

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

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

- As students encounter unfamiliar words in text, they must decompose the words by looking for patterns in syllabication and syllable types, morphemes, prefixes, and suffixes to decode the words. Students use the same patterns to encode words as they write.

Mathematics

- The introduction of input and output tables allows students to describe patterns to determine unknown values.

Science

- Students break down (decompose) an ecosystem into its smaller parts to include community, populations, organisms, and niche in order to describe how they relate to 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
4. Apply Algorithmic Thinking to Problem Solve and Create
5. Apply Computational Thinking Practices to Select, Organize, and Interpret Data

4.AP.2 The student will plan and implement algorithms that consist of sequencing, loops, variables, user input, and conditional control structures using a block-based programming language.

- a. Identify user input and its role in improving a program.
- b. Describe the concept of a variable.
- c. Read and explain a design document to trace and predict an algorithm using plain language, pseudocode, or diagrams.
- d. Create a design document to plan an algorithm using plain language, pseudocode, or diagrams.
- e. Write programs that initialize, assign values to, name, and modify variables.

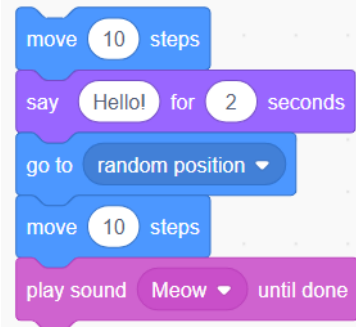
Understanding the Standard

Students apply computational thinking as they plan, create, and refine programs. This progression begins with developing foundational algorithmic thinking skills, such as sequencing steps and recognizing patterns. As students gain experience, they move into constructing and debugging simple programs, using block-based programming environments. Over time, their algorithm and programming skills expands to include more advanced concepts such as loops, conditionals, functions, and input/output operations, allowing them to design more efficient and purposeful programs.

Block-based computer programming language is a visual drag and drop programming tool that users can use to create programs using command blocks. This approach allows students to create programs with code blocks that represent programming constructs hidden from the user. The focus is on problem-solving, logical reasoning, and increasing comfort with coding concepts, which form the foundation for more advanced topics in future grades.

The use of loops to reduce redundancy and make their code more efficient has been introduced in prior grades and have allowed students the ability to handle more complex tasks with fewer instructions.

Sequencing consists of organizing steps in a logical order to complete a task. In computer science, it is the specified order in which instructions or steps are executed in an algorithm or program. It is a fundamental programming concept and refers to arranging commands or actions in a logical, step-by-step manner to achieve a desired outcome. For example, students can create a program that makes an object that moves in a sequence of steps.

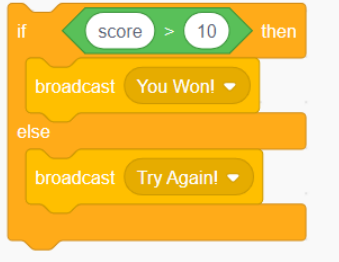
Scratch	Snap
	
Source: https://scratch.mit.edu/?scrlybrkr=eea9261a	Source: https://snap.berkeley.edu/

- Loops are a set of actions that are repeated until a certain condition is met or for a finite number of repetitions. Their purpose is to simplify code by reducing repetition and making programs more efficient. Consider this Scratch example, where instead of having to manually repeat instructions to make a character in a game move or animate multiple times, a loop allows the action to recur until a particular condition is met.

Without a Loop	With a Loop
	
Source: https://scratch.mit.edu/?scrlybrkr=eea9261a	Source: https://scratch.mit.edu/?scrlybrkr=eea9261a

- Conditional control structures use conditional logic (e.g., if-else statements) to make decisions within a computer program. These structures make decisions based upon whether or not a certain decision is true or false and allows for different actions based on certain conditions or input.

- An “If” statement executes a block of code if a statement is true. For example, if a score is greater than ten, then the program will execute the action to state that the player has won.
- An “Else” statement executes a block of code if a statement is false. For example, if a score is something “else” besides the true condition, then the program will execute the action to state that the player needs to try again.

Scratch	Snap
	
Source: https://scratch.mit.edu/?scrlybrkr=eea9261a	Source: https://snap.berkeley.edu/

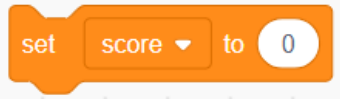
[4.AP.2a] Students build on the foundational skills developed in earlier grades by incorporating greater complexity into their block-based programming and computational thinking. This includes the introduction of user input and variables as key elements in their algorithms. By using input, students create interactive programs that respond to user choices or actions. The use of variables allows students to store, update, and reuse data throughout a program. These components enable more dynamic and flexible program design, supporting the development of logical thinking and a deeper understanding of how data flows through a program.

- User input is the data or information that a user provides to a computer program during its execution. User input allows programs to be dynamic and interactive and to respond to what the user does or types. User input can come in various forms such as text, selections from a menu, clicks, or gestures. Students recognize areas where user input is needed to improve their programs.
- A program uses input data to make decisions, update values, or interact with the user. A programmer must determine the variable and data type that will contain the user input.

[4.AP.2be]

- Variable and data types allow students to store, manage, and manipulate data.
- Variables are 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. The value of a variable can change or vary as the program runs. Key characteristics of variables include a name, a value or a type. They are analogous to containers where information can be maintained and stored or manipulated.

- On the outside of the container is a name that often adheres to naming conventions (a set of rules or guidelines for naming variables) established to ensure that code is readable, maintainable, and consistent across a program. The following example includes some sample labels that can be used for variables: userAge, x, userName, score, etc.
- Within the container is an “envelope” to store information or data that can be changed. By way of example, a user’s age, name, score, etc. Can be changed within the envelope.
- When referring to the container/envelope, the name of the container/envelope is used, not the data stored in the container.
- In the following example, students use variables within a simple game to track a player’s score.
 - Initializing a variable involves defining a variable and assigning the variable an initial value. Consider the following example where the score is set to “0.”

Scratch	Blockly
	
Source: https://scratch.mit.edu/?scrlybrkr=eea9261a	Source: https://developers.google.com/blockly

- Assigning a value to a variable involves setting a value to a variable either manually or based on program logic. Consider the following example where the score is increased by

Scratch	Blockly
	
Source: https://scratch.mit.edu/?scrlybrkr=eea9261a	Source: https://developers.google.com/blockly

- Naming a variable involves using descriptive names for variables (e.g., playerHealth, score, playerName) that make their code easier to understand. In the examples above, the variable is named “score.”
- Modifying variables involves changing their values as the program runs, often in response to user input or program events (e.g., increasing a score when a player wins a point).

- When creating a variable, it requires the user to assign it a specific data type. The data type determines how the data is processed by determining the values and operations that can be performed on that data. The data is then either stored or processed immediately and an output is provided.
- One of many operations that uses data types includes adding integers together. For example, a student creates a game, including a variable called score that holds the player's score. Since score is a whole number, it would use an integer data type.
 - `score = 0`
 - `score = score + 10` (Increases the score by 10 after a correct answer)
- Other data types include but are not limited to comparing Booleans to make decisions and displaying strings to the user. Manipulation of non-numeric data types is not expected.

[4.AP.2cd] Planning is a vital step in programming that begins with a design document. As students generate and organize ideas and algorithms using design documents (e.g., planning, drafting, revising, editing), they learn to develop and organize their algorithms. Students should be given a variety of opportunities to plan (e.g., outline, brainstorm) and draft (e.g., compose a first draft anticipating mistakes and corrections) using plain language, pseudocode, or diagrams to create their design documents. They also revise (e.g., add, remove, or rearrange ideas) and edit (e.g., proofread for grammar, punctuation, and spelling) their design documents to ensure that the algorithm is clear, sequential, and that their variables work as intended.

Design documents are a planning document that outlines the steps and/or processes required to create a product, including the problem statement, input and output requirements, and algorithmic procedures. They can include descriptions of the problem being solved, the input and output requirements, and the step-by-step processes the algorithm will follow. Design documents can be in the form of flowcharts, diagrams, or story maps that students use to brainstorm the algorithm they want to create.

- Plain language is 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.
 - In the definition of “variable” above, a plain language description is used with the bucket analogy.
- A technical explanation is, “Variables are containers for storing values, which can be changed during the program.”
- A plain language example is, “A variable is like a box where you can store something, like a number or a word. You can put different things in the box while your program is running, and you can even change what’s inside the box if needed.”

Pseudocode is simple, easy to read words that algorithms and programs are written in. It is a representation of an algorithm using a structured yet informal notation that resembles programming languages. Pseudocode allows students to write down the logic without getting bogged down by syntax rules.

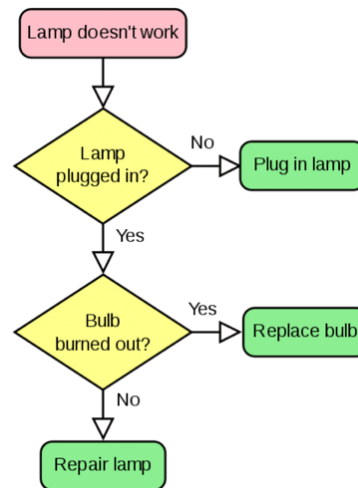
- Pseudocode is often easier to read and understand than actual code.

- The following uses the if-else conditional statement to check whether the user is old enough to drive. The program requires the user to input their age as input data. Based on the user's input the program will produce an output value that provides feedback saying they are old enough to drive or not old enough to drive.

```
START
  DISPLAY "What is your age?"
  GET user input for age
  IF age >= 16
    DISPLAY "You are old enough to drive!"
  ELSE
    DISPLAY "You are not old enough to drive yet."
  END IF
END
```

Pseudocode example adapted from commonly available online examples.

Diagrams are visual representation of data, information, or concepts. Visual representations, such as flow charts, illustrate how the algorithm works. The following diagram is a flowchart.



Students trace an algorithm through this document by following the steps outlined in the design document to see how the algorithm processes input data to produce output and to note how variables change throughout the document. After tracing the algorithm, students predict the output for given inputs (Mathematics 4.PFA.1.abc).

Concepts and Connections

CONCEPTS

Developing a foundational understanding of user input, variables, and algorithm design is essential in programming. Learning how user input influences program functionality, how variables store and modify data, and how to analyze and create design documents using plain language, pseudocode, and diagrams strengthens problem-solving skills. These concepts enable the ability to write and refine programs while recognizing multiple approaches to achieving solutions.

CONNECTIONS

Within the grade level/course: At this grade level, students continue building upon their block-based programming language by adding loops, variables, and user-input to their events and conditional control structures. They explore the concept of a variable and write programs that initialize, assign values to, name, and modify variables (4.AP.2).

Vertical Progression: In Grade 3, students used conditional control structures as they planned their algorithms (3.AP.2). In Grade 5, students will include nested conditional control structures in their algorithms (5.AP.2).

ACROSS CONTENT ALIGNMENT

English

- **4.RI.3C** The student will use textual evidence to demonstrate comprehension and build knowledge from grade-level complex informational texts read by describing the relationship between a series of historical events, scientific concepts, or steps in technical procedures using words that pertain to comparison, sequence, or cause and effect. *(Note: RI standards are only aligned if students are building CS knowledge from grade level complex informational texts)*

Mathematics

- **4.CE.1c** 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 by applying strategies (e.g., place value, properties of addition, other number relationships) and algorithms, including the standard algorithm, to determine the sum or difference of two whole numbers, where addends and minuends do not exceed 10,000.

- **4.CE.2** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using multiplication with whole numbers, and single-step problems, including those in context, using division with whole numbers; and recall with automaticity the multiplication facts through 12×12 and the corresponding division facts.
- **4.CE.3** The student will estimate, represent, solve, and justify solutions to single-step problems, including those in context, using addition and subtraction of fractions (proper, improper, and mixed numbers with like denominators of 2, 3, 4, 5, 6, 8, 10, and 12), with and without models; and solve single-step contextual problems involving multiplication of a whole number (12 or less) and a unit fraction, with models.
- **4.CE.4** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using addition and subtraction of decimals through the thousandths, with and without models.
- **4.MG.1** The student will reason mathematically to solve problems, including those in context, that involve length, weight/mass, and liquid volume using U.S. Customary and metric units.
- **4.MG.2** The student will solve single-step and multistep contextual problems involving elapsed time (limited to hours and minutes within a 12-hour period).
- **4.MG.3** The student will use multiple representations to develop and use formulas to solve problems, including those in context, involving area and perimeter limited to rectangles and squares (in both U.S. Customary and metric units).
- **4.PFA.1** The student will identify, describe, extend, and create increasing and decreasing patterns (limited to addition, subtraction, and multiplication of whole numbers), including those in context, using various representations.

Science

- **4.1** The student will demonstrate an understanding of scientific and engineering practice by a) asking questions and defining problems; b) planning and carrying out investigations; c) interpreting, analyzing, and evaluating data; e) developing and using models; f) obtaining, evaluating, and communicating information. *4.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 4.1 include: 4.2 (plants and animals), 4.3 (organisms), 4.4 (weather conditions and phenomena).*

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

- Before writing a narrative story, students will complete a story map that outlines key plot elements (characters, setting, problem, events, and resolution). Then, they will use the story map as a design document to plan the sequence of events, similar to planning an algorithm. Students will trace their planned story step-by-step to predict how the narrative will unfold, identifying cause-and-effect

relationships and checking for logical flow. This process helps students understand how a design document can be used to structure and predict outcomes in both storytelling and algorithms.

Mathematics

- Students describe the rule for input and output tables and use these rules to determine the output when given the input, and to determine the input when given the output. Consider the following example:

Input	Output
0.25	0.5
2.75	3
?	3.75
4.5	?

Rule: _____

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

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

- a. Create and test programs that consist of sequencing, loops, variables, user inputs, and conditional control structures.
- b. Create and use variables to store and process data.
- c. Trace and predict the value of variables that change over the course of the program's runtime.
- d. Analyze and describe program results to assess validity of outcomes.
- e. Revise and improve programs to resolve errors or produce desired outcomes.

Understanding the Standard

The iterative design process is a systematic approach to creating and refining products, systems, or solutions through repeated cycles of design, evaluation, and improvement. It is a key programming approach that gives students hands-on problem-solving experience. It involves a repeated cycle of designing, testing, debugging, and refining code. Through this process, students learn that programming is an ongoing and repetitive effort to improve and adapt based on feedback and results.

[4.AP.3ab] Through block-based programming, the iterative design process is applied to create and test programs, following the criteria outlined in Computer Science 4.AP.2. Building upon foundational programming concepts such as events, loops, and conditional control structures, variables are introduced with meaningful names (e.g., temperature, environmental factors, increase) to store and manipulate data dynamically. The role of variables in processing information is explored, including how their values change over time based on user input and program logic.

In programming, tracing refers to the process of following the flow of an algorithm or program step by step to track how variables change throughout execution. This is done by systematically analyzing the program's logic, either manually or using debugging tools, to predict variable values and understand how data is manipulated at different stages. Tracing allows students to predict how the value of a variable changes over time, helping them understand the dynamic nature of program execution.

- A recommended approach in tracing the flow of an algorithm or program includes the following:
 1. Read the entire program to understand its overall structure and purpose.
 2. Identify the variables used and note their initial values.
 3. Start at the first line of code and move line by line in order the program runs.
 4. Record the value of each variable as it changes after every line of execution.
 5. Follow control structures like conditionals (e.g., *if/else*) and loops, and note which path or iteration is taken based on the conditions.

6. Predict the program's output based on the variable values and logic you've traced.
7. Compare traced output to the actual result when the program runs to check for accuracy and identify any logic or syntax errors.

[4.AP.3c] Analyzing a program's results helps determine whether it works as expected and produces accurate outcomes. This process involves tracing how variables change, predicting their behavior, and checking if the program follows the intended logic. By reviewing outputs and identifying inconsistencies, adjustments can be made to improve efficiency and reliability.

- For example, in a weather simulation, students may analyze how a temperature variable evolves with each loop iteration and assess how user input modifies program flow. By interpreting these results, students determine whether the program accurately models expected behaviors, validate the logic behind variable updates, and refine their approach to ensure data integrity. This process reinforces the importance of evaluating computational outputs for correctness and reliability.

[4.AP.3d] After the execution of a program, the output should be analyzed to determine whether the results align with the expected outcomes. This process involves assessing the program's correctness, identification of errors or unexpected behaviors, such as variables not updating properly or conditions failing to respond and recognizing patterns that indicate flaws in the program's logic. This iterative process reinforces the importance of refining and improving computational solutions.

When algorithms do not function as intended, a systematic review is required to identify the issue, trace the sequence of flawed steps, and determine potential solutions. Adjustments may involve modifying variable assignments, restructuring control flow, or refining conditional statements to achieve the desired outcome. Once adjustments are identified they are implemented and tested, reinforcing the iterative design process as a method of continuous improvement in programming.

[4.AP.3e] Revising and improving an algorithm is a key part of the debugging process, where students identify and fix problems in their code to ensure it works correctly. Debugging is the process of identifying, isolating, and resolving errors, commonly known as "bugs", within a set of instructions, code, or a system. This critical skill allows students to analyze unexpected behavior, correct flaws in logic, and refine their code to ensure the program operates as intended and meets its defined objectives. Effective debugging involves systematically tracing variable changes, examining control flow, and using debugging tools to pinpoint errors.

- For example, while testing a program, students notice that a loop runs too many times, causing incorrect output. To debug the issue, they analyze the loop condition and identify that a variable controlling the number of iterations is not updating correctly. Students revise the condition to ensure the loop executes as expected, improving both accuracy and efficiency.

Concepts and Connections

CONCEPTS

Understanding how variables function in a program is essential for storing and processing data effectively. Tracing their changes throughout execution helps predict outcomes, assess program accuracy, and refine logic to improve performance. Through iterative revisions, errors are corrected, and programs are optimized to ensure reliability and meet desired goals.

CONNECTIONS

Within the grade level/course: At this grade level, students use block-based programming to explore user inputs and to create and use variables for the purpose of storing and processing data. They further explore variables by tracing and predicting the value of the included variables as part of the iterative design process (4.AP.3).

Vertical Progression: In Grade 3, students' algorithms included events, loops, and conditional control structures that they analyzed to assess the validity of outcomes and to revise and improve their programs as part of the iterative design process (3.AP.3). In Grade 5, students will use accurate terminology to describe and explain the iterative design process, create programs that include two-way branching conditional control structures, and trace and predict the outcomes of programs (5.AP.3).

ACROSS CONTENT AREAS

English

- **4.W.2A** The student will compose various works for diverse audiences and purposes, linked to grade four content and texts by engaging with writing as a process to compose well-developed paragraphs. *(Note: Students are writing multi-paragraph compositions about their programs and relying on grade level text as necessary.)*

Mathematics

- **4.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.
- **4.CE.2** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using multiplication with whole numbers, and single-step problems, including those in context, using division with whole numbers; and recall with automaticity the multiplication facts through 12×12 and the corresponding division facts.

Science

- **4.1** The student will demonstrate an understanding of scientific and engineering practice by a) asking questions and defining problems; b) planning and carrying out investigations; c) interpreting, analyzing, and evaluating data; e) developing and using models; f) obtaining, evaluating, and communicating information. *4.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 4.1 include: 4.6 (solar system)*

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

- The writing process engages students with the iterative design process. As students write well-composed paragraphs, they learn that they must include an introduction with a clear topic sentence connecting to the main idea, supporting details relevant to the topic, purpose and genre, and a concluding sentence that transitions the paragraph into the next. Upon completion of their writing, students evaluate their paragraph to ensure that the writing is clear and contains the required information necessary for developing the topic. If it does not, they revise and edit the paragraph to eliminate errors and improve clarity.

Science

- To describe the concept of a variable, students are provided with lab experiment cards containing examples of different variables they may find in an investigation into how light affects plant growth (i.e., the amount of light, the height of the plant, the amount of soil and water, etc.). Students sort the cards into three different containers labeled “independent,” “dependent,” and “constant.” Students equate scientific variables to different parts of a program that store things in a container-like format and collect 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](#).

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

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

Computing Systems (CSY)

4.CSY.1 The student will model how a computing system works to accomplish a task.

- Describe how computing systems perceive the world through sensors and other inputs.
- Compare and contrast how humans and computers process information from inputs.
- Explain how computing devices may be used to classify and organize input.
- Diagram and describe a simple computing system indicating processors, inputs, and outputs.

Understanding the Standard

A computing system is an integrated group of hardware and software that work together to store, process, and manage data. An integrated group of hardware (the physical components like the processor, memory, and input/output devices) and software (the programs and operating systems) work together in a computing system to perform various tasks.

[4.CSY.1a] Computing systems perceive the world by using sensors and other input devices such as hardware components that allow users to enter data into a computing device to collect information just like a human uses their senses to understand their surroundings. “Perception is the extraction of meaning from sensory information using knowledge.”¹

Computing systems require input and output. Input and output, also referred to as I/O, is communication between an information processing system, such as a computer, and the outside world, possibly a human or another information processing system.

- Input is data that is entered or received by a computing device for processing.

There is a wide variety of digital collection tools used for gathering and inputting digital data. Tools are chosen based upon the type of measurement they use as well as the type of data people wish to observe. These collection tools include the movements and clicks of your mouse and the keys you type on a keyboard. Sensors are also used in computing systems, to detect information and serve as input devices for the system. Sensors may include but not limited to:

- Proximity sensor – used to detect nearby objects.
- Accelerometer – used to measure changes in speed or direction.
- Light sensor – used to detect ambient light levels.
- Gyroscope sensor – measures rotation and orientation.
- Biometric sensor – capture data related to physical characteristics.
- Camera – used to capture visual images in the form of photos or videos.
- Microphone – used to convert sound waves into electrical energy variations that can then be recorded, amplified, or transmitted.

- Thermometer – used to measure temperature.
- Outputs are data or information produced by a computing device after processing input. For example, outputs include data and what can be seen on the computer screen or how the robotic device responds based on the input from the sensor.

Computers process information obtained from inputs to perform tasks requested by the user such as calculations, comparisons, or sorting. Once complete the results of the task are completed, they may be displayed on a monitor, printed on paper or in 3D, or displayed in other ways for the user.

[4.CSY.1b] Humans and computers share similarities in processing information by recognizing patterns and following logical steps, however humans also rely on prior experiences and emotions. Computers only process information logically and objectively. Humans and computers both process information, but in different ways. Humans use their senses (like sight and hearing) to gather information from the environment, then the brain interprets and responds using memory, experience, and emotions. Computers, on the other hand, take in data through input devices (like keyboards or sensors), process it using hardware and software, and produce output based on programmed instructions.

While humans rely on neurons to transmit signals and make decisions, computers use circuits and algorithms to follow step-by-step procedures. Both systems store information. Humans store information in the brain, whereas computers store information in memory or storage. Conversely, humans are better at making connections and adapting to new situations as humans can build upon prior experiences and emotions. Computers, however, are faster at calculations and precision.

Processing Information		
	Humans	Computers
Input	The information we receive from the world around us through our five senses: sight, sound, touch, taste, and smell.	This is the information or data that a computer receives from external sources. It can come from typing on a keyboard, touching a screen, clicking a mouse, or using sensors to detect things like movement or temperature.
Processing	Our brain processes the information we receive from our senses. It helps us understand what we're experiencing by analyzing the details and comparing them to things we've learned or experienced before. This allows us to make decisions, solve problems, and respond to what's happening around us.	Once the computer receives input, the Central Processing Unit (CPU) takes over to process the information. The CPU follows a set of instructions, called a program, to decide what actions to take. This might include solving problems, comparing data, or organizing information so the computer can respond correctly.

Memory (Storage)	The brain stores information, which helps us remember things over time. Some memories are short-term, like remembering directions just long enough to follow them. Others are long-term, like remembering your birthday or how to ride a bike. Memory helps learn from past experiences and make decisions based.	As the computer processes information, it may need to temporarily store data in its memory so it can use it during the task. This short-term storage helps the computer keep track of important details as it works. For example, during a game, the computer remembers your current score and level, updating this information in real time as you play. Computers also use long-term memory to save important information that needs to be available, like your saved games, documents, or pictures—even after the computer is turned off.
Response (Output)	After processing information, the brain sends signals to help us respond. This response is our “output”. It might be something we choose to do, like answering a question or waving to a friend. Some responses happen automatically, like blinking, breathing, or pulling your hand away from something hot. These outputs help us stay safe and interact with the world around us.	Once the computer finishes processing information, it produces an output, a result we can see, hear, or interact with. This output might be words or images on a screen, a printed document, a sound like music or a beep, or even a physical action, like moving a robot’s arm. The output is how the computer communicates its response to the input it received.

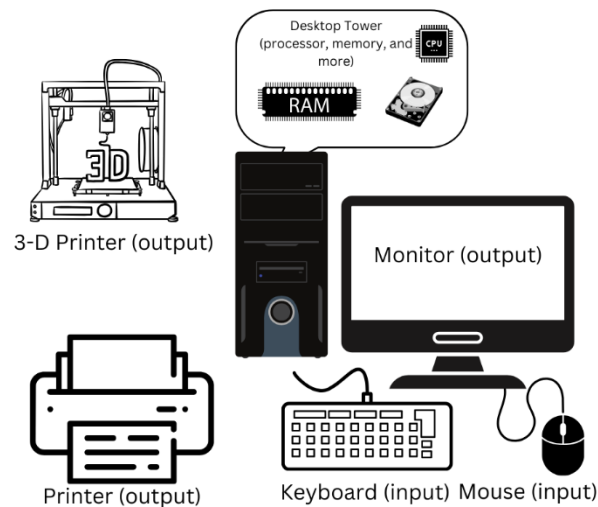
[4.CSY.1c] Computers frequently classify information in the same way that humans would classify toys or books into groups. When computers classify information, they follow an algorithm. The following table is just one example of how computers frequently classify information.

Step	Description	Example
1. Receiving Input	The computer takes in information like typing, photos, or sounds.	Uploading a bunch of animal photos.
2. Classifying	The computer puts information into groups, or categories, based on similarities or characteristics.	Sorting animal photos by type (dogs, cats, birds) or by color.
3. Organizing	It organizes information by following specific rules, like arranging by date, size, or name.	Sorting homework files by subject, date, or grade.

4. Displaying Output	The computer presents the organized information in an easy-to-read way, like in folders or lists.	Creating a chart of experiment data, showing each measurement in order, or grouping similar observations for easy comparison.
----------------------	---	---

[4.CSY.1d] All computing devices are a system which has specialized parts for each task it must complete. Students should be able to recognize the general parts of a computing system and create a diagram (Computer Science 4.AP.2c, d) or model that shows these parts. A model is a simplified representation or abstraction of a concept, object, system, or process. Key parts should include:

- Input device (keyboard, mouse, etc.).
- Processor (CPU).
- Memory for storing data (hard drive).



Concepts and Connections

CONCEPTS

Computing systems interact with the world by collecting data through sensors and other inputs, processing that information, and generating outputs based on programmed logic. Understanding how computers and humans process input differently helps highlight the strengths and

limitations of automated systems. Additionally, computing devices classify and organize input to improve efficiency, and their structure, including processors, inputs, and outputs, can be diagrammed to illustrate how data flows through a system.

CONNECTIONS

Within the grade level/course: At this grade level, students are familiar with the parts of a computing system (2.CSY.1) and are using that knowledge to understand how the parts work together to accomplish a task. Students learn about sensors and the information they can provide for a computing system. Students explore how humans and computing devices process information. Classifying and organizing information obtained from inputs are tasks that computing devices do on a regular basis.

Vertical Progression: In Grade 3, students gained an understanding of inputs and outputs (3.CSY.1). In Grade 5, students will work toward understanding how computing systems exchange information. Understanding how the system works will prepare students to understand how various types of data are stored (5.CSY.1).

DIGITAL LEARNING INTEGRATION

- **3-5.EL** 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.

Mathematics

- Students model how a computing system works by acting out the steps of a robot designed to follow commands (inputs) to solve a real-world math problem. For example, students program a robot using simple commands to calculate the total cost of items during a pretend shopping trip. Each item has a set cost, and the robot must process inputs (item names or quantities), perform operations (addition or multiplication), and produce the output (total cost).

Science

- Students will investigate how specific body parts or structures help plants and animals survive in their environments. Then, they will compare these natural structures to how computing systems use sensors (e.g., cameras, temperature or motion sensors) to gather information. Students will model both a living organism and a computing system, showing how each uses input to respond to its

surroundings. Through this activity, they will explore how both biological and digital systems process input to support survival or accomplish a task.

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

4.CSY.2 The student will apply troubleshooting strategies when a computing system is not working as intended.

- a. Identify hardware, software, and connectivity problems using accurate terminology.
- b. Apply troubleshooting strategies to address hardware, software, and connectivity problems.

Understanding the Standard

As with any system, there are times when a computing system does not function as intended. Although computing systems may differ in design and complexity, common troubleshooting strategies can be applied across various platforms and devices. Troubleshooting is a systematic process of identifying, diagnosing, and resolving issues within a computing system. It involves applying logical reasoning, testing possible solutions, and refining approaches based on outcomes. Through consistent practice, students develop problem-solving skills, strengthen their technical understanding, and build confidence in working with computing systems.

[4.CSY.2a] Utilizing correct terminology for hardware, software, and connectivity problems is important for identifying the problem. The strategies utilized in identifying (and later fixing) the problem should involve protocols that utilize systematic protocols or procedures. These procedures frequently include if / then processes to help determine the problem (and the solution).

- Computer hardware is the physical components of a computing device that you can touch, such as the processor, memory, keyboard, and display.
- Computer software is a set of instructions that tells the computer how to act and respond, but it cannot be seen or touched. Connectivity is when different devices, like computers, tablets, or phones, can talk to each other and share information.

[4.CSY.2b] Example Hardware Troubleshooting Protocol:

- Check power and connections (power source is plugged in, ensure the power is turned on, ensure the screen is turned on).
- Check cables and connections (determine if cords completely are attached, remove extra devices, for portable devices verify the battery level is installed and charged).
- Restart the computer (shut down the computer and then turn it back on).
- Listen for sounds (listen for noises, such as beep, clicks, or fans).
- Check basic settings (view the settings to check the screen brightness, volume, Internet connection).

Example Software Troubleshooting Protocol:

- Identify the problem (software or app name and error messages).
- Check Internet connection (determine if other devices connected to the Internet without issue).

- Restart the software or app (shut down the app or software and then restart).
- Restart the computer (shut down the computer and then turn it back on).
- Check for updates (view the settings or system menu to check for updates).
- Check for issues on other devices using the same program or app (if other devices are working, check settings for potential updates).

Example Connectivity Troubleshooting Protocol:

- Check the Wi-Fi Connection (reconnect if needed).
- Check Airplane Mode (this should be turned off).
- Restart the device (power off the device completely and wait a few minutes before restarting).
- Check other websites or apps (try accessing different apps or websites, if they are working, check status on another device).
- Check for issue on other devices using the same Wi-Fi connection (use another device to test if connection works, if it does, then the problem is the original device).
- Move closer to Wi-Fi (signal strength weakens with distance and interference from walls or other devices).
- Disconnect and Reconnect to the network (in settings forget the network and reconnect and enter password).

Concepts and Connections

CONCEPTS

Recognizing and diagnosing hardware, software, and connectivity issues is essential for maintaining reliable computing systems. Understanding technical terminology allows for precise identification of problems, while applying troubleshooting strategies—such as testing connections, adjusting settings, or replacing faulty components—helps resolve issues efficiently. These skills ensure devices function correctly and remain operational in various environments.

CONNECTIONS

Within the grade level/course: At this grade level, students continue to strengthen their troubleshooting skills as they engage with increasingly complex computing systems. While computing systems may not always function as expected, applying effective troubleshooting strategies enables students to work more independently and confidently when resolving issues (4.CSY.2).

Vertical Progression: In Grade 3, students used correct terminology for when a computing system does not work as expected and applied troubleshooting strategies for common issues (3.CSY.2). In Grade 5, students work toward being able to apply troubleshooting skills to identify new and different problems they have yet to experience. Students use computational thinking strategies when identifying a problem and following protocols to fix the problem. (5.CSY.3).

ACROSS CONTENT AREAS

English

- **4.W** The student will compose various works for diverse audiences and purposes, linked to grade four content and texts. *(Note: Students are writing multiparagraph compositions about their programs and relying on grade level text as necessary)*

Mathematics

- **4.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.
- **4.CE.2** The student will estimate, represent, solve, and justify solutions to single-step and multistep problems, including those in context, using multiplication with whole numbers, and single-step problems, including those in context, using division with whole numbers; and recall with automaticity the multiplication facts through 12×12 and the corresponding division facts.

DIGITAL LEARNING INTEGRATION

- **3-5.EL** Students leverage technologies, including assistive technologies, to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.
D. Understand the various fundamental concepts of technology operations, demonstrate the ability to choose, use, and troubleshoot technologies and transfer knowledge to explore emerging technologies.
- **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.
B. Use collaborative technologies to work with others, including peers, experts, and community members to examine issues and problems from multiple viewpoints.

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 create quick reference guides to help classmates solve common computing device problems they may face at school or home. Peers will test the guides by following the steps and evaluating how clear and helpful they are. Students may then work in pairs to revise and improve any unclear instructions, practicing debugging and effective communication. Final guides can be compiled into a class resource book or shared with other students across the school

Mathematics

- Students will compare troubleshooting techniques used in computing to problem-solving strategies in mathematics. Just as programmers use debugging to identify and correct errors in code, students will analyze and revise steps in math equations to correct mistakes and reach accurate solutions. They will explore how both processes involve logical reasoning, step-by-step analysis, and revision.

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

4.CSY.3 The student will describe the learning process of humans and computers.

- a. Compare and contrast how humans and computing technologies collect, store, and process data.
- b. Identify similarities and differences on how humans and computing technologies infer and extract meaning from data.
- c. Define machine learning and identify machine learning approaches: supervised, unsupervised, and reinforcement learning.

Understanding the Standard

Humans learn by observing, practicing, making mistakes, and applying prior knowledge to new situations, often influenced by curiosity and social interaction. Computers, especially those using machine learning, learn by processing large amounts of data to identify patterns and adjust their output based on programmed criteria. Unlike humans, computers require clear instructions and data to learn, while humans can learn from unstructured experiences and adapt to changing environments.

[4.CSY.3a] Humans and computing technologies both collect, store, and process data, but they do so in different ways. Humans gather data through their senses, such as sight and hearing, while computers use input devices like sensors, cameras, and keyboards. Human brains store information as memories shaped by experience and emotion, whereas computers store data digitally using hardware like hard drives or cloud storage. When processing data, humans rely on reasoning, intuition, and context, allowing for flexible thinking, while computers follow precise algorithms to process data quickly and consistently. Overall, humans are better at interpreting meaning and adapting to new situations, while computers excel at handling large amounts of structured data efficiently.

- Humans can learn by taking in information (input - often the five senses), processing information (thinking), and making decisions (output). When humans process information, they also make connections to prior experiences and emotions. Humans can improve their skills with practice.
- Computers process information by taking in information (input – keyboard, mouse, microphone, sensors). The processor executes instructions (performing calculations or logic operations), the memory stores data and instructions temporarily while tasks are processed, and processed data is sent to an output device (monitor, speakers, printer).

[4.CSY.3b] Computers need exact step-by-step directions to learn something new, typically through algorithms that guide their processing. They can also recognize patterns in large amounts of information using techniques such as machine learning, where the system improves performance based on data input and feedback. Unlike humans, computers do not learn intuitively; they rely on structured data and predefined models. Emotion and prior experiences do not impact how computers learn, as their learning is not influenced by personal context. Instead, their ability to improve depends on the quality, quantity, and variety of data they receive.

When humans are given data and asked to infer or extract meanings, they are using clues to understand things that are not directly said. They may also utilize prior knowledge or emotions when making inferences. Computers rely on patterns or algorithms to make inferences because

they do not have prior experiences or emotions. Both use data and patterns to find meaning and make sense of the information. The big difference is humans use intuition and experiences while computers can only follow precise instructions.

[4.CSY.3b] Computers utilize machine learning. Machine learning is an approach that occurs when computers learn from examples instead of following exact instructions. This allows computers to improve their performance over time by identifying patterns in data and making predictions or decisions without being explicitly programmed for every task.

Machine learning approaches include supervised, unsupervised, and reinforcement learning.

- Supervised learning is when a computer learns to make decisions or predictions by looking at examples that have the correct answers already provided.
- Unsupervised learning is when a computer learns from data without being told what the answers are. It is when a computer learns to find groups or patterns without anyone telling it what is right or wrong.
- Reinforcement learning involves teaching a computer through trial and error, where it learns by getting rewards or punishments for the actions it takes.

Concepts and Connections

CONCEPTS

Computing systems gather, store, and process data in ways that differ from human cognition. While humans rely on context and reasoning, computers utilize algorithms to extract and interpret information efficiently. Machine learning further enhances this ability by enabling systems to recognize patterns, improve predictions, and adapt through techniques such as supervised, unsupervised, and reinforcement learning.

CONNECTIONS

Within the grade level/course: At this grade level, students are utilizing computational thinking skills as they compare and contrast the learning process of humans and computing technologies. This can be connected to generative artificial intelligence.*

Vertical Progression: In Grade 5, students will begin to explore automated decision-making practices utilized by computing systems (5.CSY.2).

ACROSS CONTENT AREAS

English

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

DIGITAL LEARNING INTEGRATION

- **3-5.EL** Students leverage technologies, including assistive technologies, to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.
D. Understand the various fundamental concepts of technology operations, demonstrate the ability to choose, use, and troubleshoot technologies and transfer knowledge to explore emerging technologies.
- **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.

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 conduct research to compare how humans and computing systems learn and process new information. Using credible sources, they will gather and organize information, then write an informational text explaining the similarities and differences in how the brain and machines handle input, recognize patterns, and improve over time.

Science

- Students will collect temperature data from different areas of the school using three types of tools: traditional thermometers, digital thermometers, and sensors connected to computing devices. In groups, they will choose how to display their results, then compare how different visualizations affect interpretation. After guidance on selecting a common graph type, all groups will display their data in the same format and analyze differences in results, discussing factors that may have influenced 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, Communication, and Document Solutions
4. Test and Optimize Artifacts

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

Cybersecurity (CYB)

4.CYB.1 The student will examine the impacts of appropriate and inappropriate use of computing technologies.

- a. Examine and explain scenarios for appropriate and inappropriate use of computing technologies.
- b. Develop possible solutions involving inappropriate use of computing technologies.

Understanding the Standard

[4.CYB.1a] Computer networks, including the Internet, can be used to connect people to other people, places, information, and ideas. In order to keep students safe, schools have rules and guidelines that define safe practices and responsible use of technology commonly known as the Acceptable Use Policy (AUP). Students are aware of what is allowed and not allowed when using technology and know there are consequences for violating school / division policy. Students should know how to report behaviors that violate the AUP to school officials.

As students increase their use of the networks and interact with others outside of the school or home environment, digital safety is an increasing concern. It is the job of all staff to monitor students when engaged with technology.

Appropriate Use of Technology Scenarios Examples:

- Using a school device to research historical events that resulted in the settlement of Virginia.
- Collaborate with a team of students to create a slide presentation about Virginia's natural resources.
- Practice math or reading skills utilizing district purchased resources.

Inappropriate Use of Technology Scenarios Examples:

- Visiting websites that are not approved by school / district.
- Using school communication platform (email or chat) to send mean or threatening messages to classmates or school officials.
- Playing games on websites not approved for students by school or district.

Concepts and Connections

CONCEPTS

Computing technologies can be used responsibly to enhance productivity, communication, and problem-solving, but they can also be misused in ways that lead to security risks or ethical concerns. Understanding the difference between appropriate and inappropriate use helps ensure safe and effective interactions with technology. When misuse occurs, such as unauthorized data access or cyberbullying, solutions may involve implementing security measures, enforcing digital policies, or promoting responsible online behavior to mitigate risks and protect users.

CONNECTIONS

Within the grade level/course: At this grade level, students focus on appropriate and inappropriate use of computing technologies (with a focus in the school setting). Students also learn that there are consequences for not following rules in the computing world, just like other aspects of life (4.CYB.1).

Vertical Progression: In Grade 3, students learned about the safe use of computing technologies (3.CYB.1). In Grade 5, students will learn about ways to limit unauthorized access on computing devices (5.CYB.1).

ACROSS CONTENT AREAS

History and Social Science

- **VS.h** The student will apply history and social science skills to the content by practicing civility, respect, hard work, honesty, trustworthiness, and responsible citizenship skills.

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.
- **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.

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 work in small groups to create digital safety guides that promote respectful and responsible behavior when using computing technologies. They will examine scenarios involving appropriate and inappropriate online behavior (e.g., sharing passwords, cyberbullying, citing sources, or using kind language), and relate these to civic values such as honesty, trustworthiness, and respect. Each group will use a

visual format—such as a slide deck, infographic, or interactive story—to show the impact of digital choices and provide tips for practicing digital 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. Evaluate Resources and Recognize Contributions

4.CYB.2 The student will identify and investigate best practices to safeguard information shared online and through online platforms.

- a. Classify personal, private, and public information.
- b. Research and evaluate tradeoffs of sharing information.
- c. Investigate and communicate best practices to limit unauthorized access to information on a computing device.
- d. Demonstrate proper use and protection of personal passwords.
- e. List methods used to safeguard online information.

Understanding the Standard

[4.CYB.2] Connecting devices to a network or the Internet provides benefits to the user, but there are risks involved. When devices are connected to the Internet, information is shared with others. Sometimes information is shared directly, like when a form is filled out to register for an event. Other times information is shared indirectly through cookies and data tracking. Information may be classified into personal, private, and public information. Information that is personal should be protected and not shared publicly. Understanding the differences in the types of information and why the information is being collected is critical to being safe online.

Sharing information requires the knowledge that not all information is the same, the ability to classify information, and critical thinking to determine if the information should be shared. Learning to evaluate why the information is being shared helps students become better digital citizens.

[4.CYB.2a]

Types of Information		
Personal	Private	Public
Personal information is data or information about a person that relates to their identity, characteristics, or activities. It includes any detail about you that makes you unique but doesn't directly identify you or put your safety at risk.	Private information includes sensitive data or information that can identify a person or give others access to your personal life.	Public information is information that's okay to share with anyone and is typically available for everyone to see.

• First name only • Favorite color • Favorite sport • Hobby you enjoy • Pet name	• Full name • Home address • Birthday • School name • passwords	• School assignments • General facts (the Earth orbits the sun) • General school rules • School spirit days
--	---	--

[4.CYB.2b] Sharing personal, private information can provide opportunities of convenience, like personalized services and social connections. However, sharing information also comes with risk, such as misuse of data by companies or hackers. It is important to evaluate the tradeoffs of sharing personal, private information.

[4.CYB.2c] Unauthorized access to information on a computing device is the ability for someone to view, use, or take data without permission from the device owner. This can include hacking, installation of malware, and/or physical access to a computing device. Limiting unauthorized access to information on a computing device can help reduce data theft and protect private information. Unauthorized access is when information is accessed without the permission of the owner. Best practices to limit unauthorized access include creating strong passwords, keeping software and operating system up to date, logging out of a device when finished, not sharing private information online, and keeping your device in a safe location.

[4.CYB.2d] Passwords are a secret word or phrase used to protect devices and information from unauthorized access. Passwords contain a combination of characters used to verify a user’s identity and protect access to devices, accounts, or information. It acts as a key to ensure that only authorized individuals can gain entry. There are computer programs that can be used to decode passwords; therefore, strong passwords have characteristics that make them more difficult to guess. To enhance security, many systems enforce password policies that specify requirements for length, complexity, and character variety, helping users generate stronger passwords.

Suggestions for creating strong passwords for students include:

- Use uppercase and lowercase letters.
- Use numbers.
- Use symbols.
- Use at least 8 characters.
- Don't use words from a dictionary.
- Don't use the same password twice.
- Don't use personal information.

Best practice is to emphasize the importance of not sharing passwords both in the classroom and at home. Entering passwords on only trusted websites is also crucial to password safety.

[4.CYB.2e] The ability to safeguard online information involves implementing measures to protect online data from unauthorized access, use, or disclosure. There are several ways to safeguard online information from unauthorized access.

- Utilizing strong passwords.
- Encrypting data
- Keeping private information private.
- Being cautious of links and emails from unknown sources.
- Using secure websites.
- Log out of accounts.
- Updating software regularly.

These safeguard practices help maintain confidentiality, integrity, and availability of information in digital environment.

Concepts and Connections

CONCEPTS

Sharing information online comes with tradeoffs, balancing accessibility and privacy while considering potential risks. Evaluating these tradeoffs helps ensure responsible data management and security. Best practices for limiting unauthorized access include using strong passwords, enabling multi-factor authentication, and securing devices against cyber threats. Proper password management and safeguarding online information through encryption, firewalls, and secure browsing habits further enhance digital security.

CONNECTIONS

Within the grade level/course: At this grade level, students learn to classify personal, private, and public information. The use of strong passwords is only one of the methods of safeguarding information online (4.CYB.2).

Vertical Progression: In Grade 3, students explored practices to keep information safe and created and used strong passwords (3.CYB.1). In Grade 5, students will learn about the consequences of unauthorized access to devices (5.CYB.1).

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.

History and Social Science

- Students research Virginia figures from the Revolutionary War, such as George Washington or Patrick Henry, and examine how important communication and trust were in leading and protecting information during conflict. Students will create infographics that show the difference between private and public information, connecting historical practices of safeguarding messages to modern cybersecurity habits.

Mathematics

- Students will collect data from classmates and staff regarding password safety practices, including topics such as password strength, reuse across multiple sites, and sharing habits. They will organize and display the data using appropriate graphs or charts. After analyzing the results, students will identify areas where additional training in online safety may be needed. Based on their findings, they will design and deliver multimedia presentations to educate others on best practices for creating and protecting strong 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. Evaluate Resources and Recognize Contributions

4.CYB.3 The student will examine how information is shared online and explain the importance of cybersecurity.

- Investigate multiple ways people share information online.
- Determine and describe when information should be shared and to whom it should be shared.
- Describe how personal information can be collected and shared online.
- Explain the importance of cybersecurity.

Understanding the Standard

[4CYB.3a] People share information online in multiple ways including but not limited to email, blogs, and websites. Information can be shared online for many purposes including, but not limited to communication, expression of creativity, education and learning, and entertainment.

[4.CYB.3b] Determining when and with whom information should be shared requires understanding the difference between personal, private, and public information can help users determine what information to share and whom to share it with. (Refer to Computer Science 4.CYB.2) Evaluating the purpose behind the information request helps ensure that sharing is appropriate and necessary. Considering potential risks is an everyday practice of responsible decision making.

Students should not provide private information without parental consent. (Refer to Computer Science 4.CYB.2)

Understanding that not all information should be shared with everyone. Public information may be safe to share because it is available to the public. Personal information is usually safe to share in most settings because it does not provide enough information to identify the individual. Private information can identify the individual and should only be shared with trusted sites or individuals.

To Share or Not to Share?	
Safe to Share	Not Safe to Share
- First name - Favorite hobby - Respectful opinions	- Full name - Home address - Phone number

[4.CYB.3c] Personal information can be collected in various ways such as but not limited to signing up for websites, filling out online forms, and through social media interactions. This information can sometimes be shared with advertisers or other companies.

Information is sometimes collected and shared without a person's knowledge or permission. For example, user data such as location data, IP Address, browsing history, and personal information may be collected when using public Wi-Fi. Wi-Fi is 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.

Phishing scams are a deceptive online attack where scammers pretend to be a trusted source and trick individuals to share personal identifiable information, like passwords or credit card numbers. Here's how it usually works:

1. **Fake Emails or Messages:** You might receive an email or message that looks like it's from a trusted company, such as your bank, a popular website, or even a colleague.
2. **Tricking You with Links:** These messages often have urgent warnings or tempting offers, telling you to click on a link or download an attachment.
3. **Stealing Information:** When you click the link, you might be taken to a fake website that looks real. This website will ask you to enter sensitive information, which the scammers then steal. For example, an email might say, "Your account has been compromised! Click here to reset your password." The link then takes you to a fake website designed to look like your bank's login page.

[4.CYB.3d] Cybersecurity is the protection of data and information on networks and computing devices from unauthorized access, attacks, damage, or theft. As reliance and embedding of technology grows, cybersecurity has become increasingly important as it helps keep personal information safe from people who might want to misuse it. Without strong cybersecurity practices, individuals and organizations are at greater risk of data breaches, identity theft, and other cyber threats.

Concepts and Connections

CONCEPTS

Information is shared online through various methods, including social media, email, cloud storage, and collaborative platforms. Understanding when and with whom to share information is essential for maintaining privacy and security. Personal data can be collected and distributed through websites, apps, and online interactions, making cybersecurity crucial for protecting sensitive information from unauthorized access and misuse.

CONNECTIONS

Within this grade level: At this grade level, students explore how information is shared in a wide variety of ways and can be done with and without permission. Recognizing information that is safe to share and not safe to share will be beneficial when exploring how information may be collected from online sources.

Vertical progression: In Grade 3, students learned the definition of cybersecurity and how to explain problems and consequences related to inappropriate use of computing devices and networks (3.CYB.3). In Grade 5, students will explore laws and policies related to cybersecurity (5.CYB.2).

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.
- **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. Choose the appropriate technologies and resources for meeting the desired objectives of their creation or communication.

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 conduct research to learn how information was shared during the time of George Washington (e.g., handwritten letters, messengers, newspapers) and compare it to how information is shared today (e.g., email, social media, digital news). Students will gather facts from multiple sources, organize their findings, and write a short comparison piece. Students will also make inferences about potential factors such as technology, speed, or accessibility that may have caused these changes over time.

History and Social Science

- Students create a timeline showing the change in how information is safeguarded from wax seals and cyphers to currently available types of encryptions like AES-256 that is used by military, government, and financial institutions. Discuss how technological advances have impacted on how we keep our information secure.

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 Cybersecurity \(CYB\)](#)

Data and Analysis (DA)

4.DA.1 The student will identify the appropriate type of data needed to solve a problem or answer a question.

- a. Analyze a problem to determine the appropriate type of data needed.
- b. Evaluate the reliability of data sources.
- c. Use numeric values to represent non-numeric ideas to include binary, American Standard Code for Information Interchange (ASCII), and RGB values.
- d. Collect, store, clean, and organize data for analysis and to prepare visualizations.

Understanding the Standard

Data is individual pieces of information about people, things, or events that can be processed, stored, and analyzed by computing devices. The primary purpose of collecting data is to answer questions (e.g., “What brand of cell phone do fourth graders prefer?”, “What type of weather does a particular region experience?” or “What is the fourth grades’ most popular lunch choice?”). The primary purpose of interpreting data is to inform decisions (e.g., which app to download onto a phone, which type of clothing to pack for a trip based on a weather graph or which type of lunch to serve based upon class favorites) (Mathematics 4.PS.1bd).

[4.DA.1a] Data analysis is a skill that helps students to develop critical thinking, problem-solving, and analytical thinking skills that are essential in academic contexts and in everyday life. Earlier grades focus on learning to use computers to collect, store, manipulate, and organize data, to identify patterns in data, and use tools like charts or graphs to present data visually. Whereas the emphasis is on the importance of data analysis to organize data and use it to help make informed decisions and predictions, greater complexity is introduced as students analyze a problem in order to identify the correct type of data required to solve a problem or answer a question.

There are two types of data that students can collect:

- Categorical (non-numerical) data (e.g., qualitative) are observations about characteristics that can be sorted into groups or categories. It may include photos, text images, videos, non-numerical values, survey responses, true/false values, or colors.
- Numerical data (e.g., quantitative) are values or observations that can be measured. It may include heights, temperatures, scores or grades, or statistics.

The teacher can provide data sets to students in addition to students engaging in their own data collection or acquisition. Data may be acquired from resources that are already created. Consider the following examples of data collection:

- Amount of screen time each evening over the course of a month
- Number of fourth graders who own a cell phone or brand of cell phones owned by fourth graders over a five-year period
- Preferred social media platform of various age ranges over a ten-year period

- Weather data (e.g., temperature, precipitation amounts) during the fall season
- Population data over the course of five years, etc. (Mathematics 4.PS.1b; Science 4.1a)

[4.DA.1b] Not all data sources are equally reliable (i.e., a well-known weather organization versus an opinion blog), and students must be able to think critically about where information comes from, how it is collected, and whether it is supported by evidence or expert opinion so that they can make informed decisions and draw accurate conclusions. Students should be able to use search terms effectively as they collect information from multiple digital and print sources. They should also evaluate each source for relevance, validity, and credibility as they gather and synthesize information. When students consider a larger variety of sources and gauge the quality of information provided, they are able to determine which information will help them develop their projects (English 4.R.BC; Science 4.1c).

Students can evaluate the reliability of data sources using the following criteria:

- Source
- Methodology
- Consistency among other sources
- Citations and References
- Age of the data
- Purpose
- Peer Reviews
- Consistency and Accuracy

[4.DA.1c] Similar to humans, computers need a format in which to receive, interpret, and manipulate information. Computers use numeric values to store information and perform operations. When a user inputs either qualitative or quantitative data into a computer, the data must be converted into a numeric form that the computer can understand. The computer processes the data and converts it back into an output that the user can understand. This output may be in the form of words, images, videos, or sounds. Examples of different ways non-numeric or qualitative data such as photos, videos, text, or colors can be expressed include the use of different protocols such as binary, American Standard Code for Information Interchange (ASCII), or RGB values. Students are not expected to apply these protocols in fourth grade.

- Binary-- typically refers to the binary number system, which uses only two digits: 0 and 1. This concept is often introduced when teaching students about different number systems, especially in relation to computing and digital technology.
- American Standard Code for Information Interchange (ASCII) - A character encoding standard that represents text in numeric form, enabling multiple data representations in computing devices. This helps computers understand and display text. For example, the letter "A" is represented by the number 65 in ASCII.
- RGB - a color model used in digital graphics. RGB stands for Red, Green, and Blue, which are the three primary colors of light. Each color is represented by a value ranging from 0 to 255, indicating the intensity of that color.

The data cycle involves collecting data from multiple sources, securely and systematically storing it, and cleaning the data by correcting errors, removing duplicates, and imputing missing values to ensure accuracy. Once prepared, the data is organized and used to generate visualizations that reveal patterns, trends, and insights. These stages are key components of the broader data cycle, which also includes additional steps such as analysis and interpretation.

- Collecting data involves gathering the appropriate type of data needed. Students can collect data by creating a simple form or an online survey where students can type their answers.
- Storing data involves storing the data in a file or a database such as a spreadsheet.
- Cleaning data involves making sure that the data is correct, consistent, and clear. This may involve running a data-cleaning program that checks to make sure there is consistency among the data.
- Organizing data involves sorting the data into categories or using a computer program to count the number of times a category is used. This enables the students to analyze their data and use it to prepare to create graphs (bar graphs, line charts) to visualize the results of their investigations (Science 4.1c).

Concepts and Connections

CONCEPTS

Effectively analyzing a problem requires identifying the appropriate type of data needed to support decision-making and problem-solving. Evaluating the reliability of data sources ensures accuracy and credibility in analysis. Numeric values can represent non-numeric concepts, such as binary for computing operations, ASCII for text encoding, and RGB for color representation. Proper data collection, storage, cleaning, and organization are essential for preparing meaningful visualizations that enhance understanding and interpretation.

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. They 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).

Vertical progression: In Grade 3, students analyzed data to formulate questions, collected and organized data, identified problems that impact the predictions and results of the data, and used their data to make predictions and draw conclusions (3.DA.1). In Grade 5, students will identify accurate ways to collect data, organize data based on patterns, and compare and contrast the various data elements to solve a problem or investigate a topic (5.DA.1).

ACROSS CONTENT AREAS

English

- **4.R.** The student will conduct research and read a series of conceptually related texts on selected topics to build knowledge on grade-four content and texts, solve problems, and support cross-curricular learning.
- **4.R.1** Evaluation and Synthesis of Information A.) Construct and formulate questions about a topic; B) Identify search terms to location information on the topic and gather relevant information from various print and digital sources; C) Organize and synthesize information from the print and digital resources determining the relevance and reliability of the information gathered; D.) Develop notes that include important concept, summaries, and identification of information sources. *(Note: Research is based on students reading a series of conceptually related text on the CS topics)*

Mathematics

- **4.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 line graphs.

Science

- **4.1** The student will demonstrate an understanding of scientific and engineering practices by a) interpreting, analyzing, and evaluating data; b) planning and carrying out investigations; *4.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 4.1 include: 4.2 (plant and animal structures), 4.4 (weather conditions and phenomena affect), 4.7 (ocean environments), 4.8 (Virginia natural resources).*

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. Formulate problem definitions suited for technology assisted methods such as data analysis, modeling and algorithmic thinking in exploring and finding 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. Plan and employ effective research strategies to locate information and other digital sources for their intellectual or creative pursuits. 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 distinguish between categorical (qualitative) and numerical (quantitative) data that changes over time. For example, “What is your favorite lunch option?” involves categorical data, while “How many school lunches are purchased over a month?” involves numerical data collected over time. Students will then identify a question requiring data collection over time (e.g., average daily precipitation over a month, weekly plant growth, or a town's population over ten years). They will collect and organize the data in a spreadsheet or database, sort it into categories, and prepare it for creating line graphs.

Science

- Students collect and record weather data over time using instruments such as thermometers, barometers, rain gauges, and anemometers, along with sky observations. They organize the data in a spreadsheet and analyze it to select the most effective visual representation, such as a line graph or bar chart.
- Students will explore how we see color through light reflection and investigate how different materials absorb or reflect red, green, and blue light. They will then learn how digital devices use the RGB color model to represent colors by mixing light in different intensities. Using a digital tool, students will adjust RGB values to create specific colors and connect this to how our eyes perceive light. To conclude, students will reflect on how both natural and digital systems use light to produce color, and how colors can be represented numerically in computing to communicate visual information.

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

4.DA.2 The student will create and evaluate data representations to make predictions and conclusions.

- a. Formulate questions that require the collection or acquisition of data.
- b. Collect data to create charts and graphs.
- c. Recognize and analyze patterns and relationships within data sets.
- d. Analyze visual representations to make predictions and draw conclusions.

Understanding the Standard

Creating and evaluating data representations to make predictions and conclusions helps to develop data analysis skills. The process begins as students implement and reflect throughout the full data cycle. The data cycle involves formulating questions to be explored with data, collecting or acquiring data, organizing and representing data, and analyzing and communicating results (Mathematics 4.PS.1).

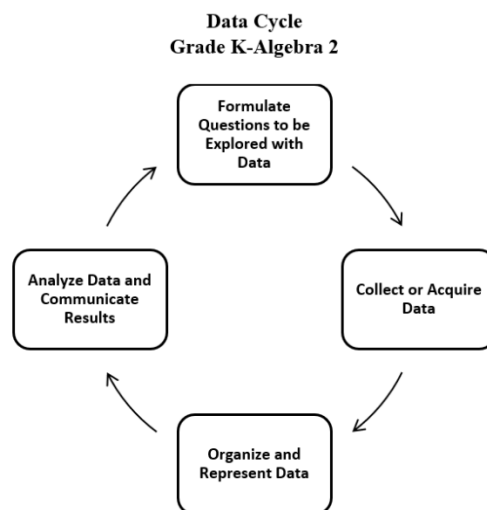


Image from: Virginia Department of Education Mathematics Standards of Learning Instructional Guide © 2024: Grade 4 [August 2024]

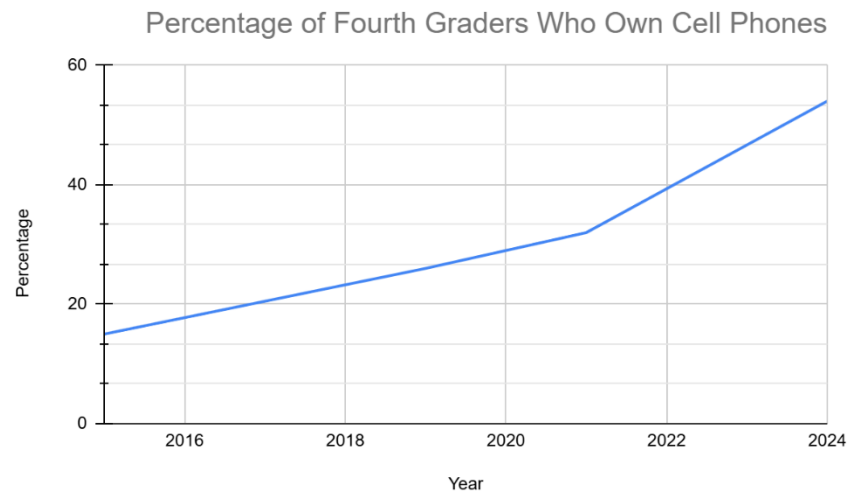
[4.DA.2a] The data cycle begins with teaching students how to identify what information is needed so that they can formulate real world questions that become manageable data collection tasks. Statistical investigations should be active, with students formulating questions about something in their environment and determining ways to answer the questions (Mathematics 4.PS.1a).

The use of open-ended questions allows for exploration and requires data collection to answer them. Questions students can ask to collect data include, but are not limited to, the following:

- How do different search algorithms perform on the same dataset?
- How does the frequency of user input affect the responsiveness of a program?
- How can data visualization techniques help us better understand trends in a dataset?
- How does the number of iterations in an algorithm affect its overall execution time?

[4.DA.2b] The data cycle involves collecting and organizing data and determining how to prepare it for visual representation. The teacher can provide data sets to students in addition to students engaging in their own data collection or acquisition (Computer Science 4.DA.2a).

- At this point, students benefit from exploring previously taught graphical representations when justifying how to best represent data clearly and accurately. Comparing different types of representations (charts and graphs) provides an opportunity to learn how different graphs can show different things about the same data. Discussions around what information different graphs representing the same data provides is beneficial for determining which graphical representation best represents the data (Mathematics 4.PS.1d).
- Technology tools (e.g., graphing software, spreadsheets) can be used to collect, organize, and visualize data. These tools support progression to analysis of data in a more efficient manner. Students are expected to engage with data that changes over time and can be represented in line graphs. A line graph of a data set shows when change is increasing, decreasing, or staying the same over short intervals of time and over the entire period of time (Mathematics 4PS.1cd).



Line Graph created with Canva

[4.DA.2c] Data analysis includes opportunities to describe the data, recognize patterns or trends, and make predictions. Once students create visual representations of data, the emphasis is on the data analysis and communication of the analysis. Students are interested in and notice individual data points and are able to describe parts of the data such as where their own data falls on the graph, which value occurs most frequently, and which values are the largest and smallest. It is important to develop student understanding as they begin to think about the set of data as a whole (Mathematics 4.PS.1d).

Students interpret data by making observations from line graphs or other visual representations by describing the characteristics of the data and the data as a whole. Consider the following examples:

- The time period when the percentage of fourth graders who owned cell phones grew the most .
- Similarities and differences in the data.
- The total percentage over a period of time.
- The time period during which no change occurred.
- The overall trend of the data.

In the line graph above, examining the line graph from left to right shows how one variable changes over time and reveals trends or progress of change in the data collected over time (Mathematics 4.PS.1d).

[4.DA.2d] The last step in the data cycle is to communicate the results of the data analysis. Students communicate their results in the form of their predictions and conclusions based off of the data. Statements representing an analysis and interpretation of the characteristics of the data should be included (e.g., patterns or trends of increase and/or decrease, and least and greatest data value). Some questions and sentence frames to guide students' predictions and conclusions in the line graph above include the following (Mathematics 4.PS.1d).

- What do you predict the percentage will be in the following year?
- Draw a conclusion about why the percentages in 2016 were lower than in 2024.
- Based on the data, what is a conclusion you can make about the ages of cell phone users over time?

Concepts and Connections

CONCEPTS

Formulating thoughtful questions guides the process of gathering and organizing data for analysis. Once collected, data can be presented in charts and graphs to highlight patterns and relationships. By examining these visual representations, meaningful trends emerge, enabling predictions and informed decision-making.

CONNECTIONS

Within this grade level: At this grade level, students expand their data analysis skills by formulating questions that require data collection, collecting data, recognizing and analyzing patterns, and analyzing visual representations so that they can make predictions and draw conclusions based on the data (4.DA.2).

Vertical progression: In Grade 3, students created and evaluated their data representations based upon data collection and analyzed it to identify patterns, draw conclusions and make predictions (3.DA.2). In Grade 5, students will analyze and use their data to create charts, graphs, and models so that they can make predictions, draw conclusions, and propose solutions to problems or questions based upon the data analysis (5.DA.2).

ACROSS CONTENT AREAS

English

- **4.R.1A** The student will conduct research and read a series of conceptually related texts on selected topics to build knowledge on grade-four content and texts, solve problems, and support cross-curricular learning by constructing and formulating questions about a topic.
- **4.R.1B** The student will conduct research and read a series of conceptually related texts on selected topics to build knowledge on grade-four content and texts, solve problems, and support cross-curricular learning by identifying search terms to locate information on the topic and gather relevant information from various print and digital resources. *(Note: English RI standards are only connected if students are reading grade-level informational text and then applying the standards to this informational text.)*

History and Social Science

- **VS.c** The student will apply history and social science skills to the content by developing questions, enhancing curiosity, and engaging in critical thinking and analysis.

Mathematics

- **4.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 line graphs.
- **4.PS.2** The student will model and determine the probability of an outcome of a simple event.

Science

- **4.1** The student will demonstrate an understanding of scientific and engineering practices by c) interpreting, analyzing, and evaluating data; e) developing and using models; f) obtaining, evaluating, and communicating information; *4.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 4.1 include: 4.4 (plant and animal structures)*

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.
- **3-5.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.

Mathematics

- Students will analyze line graphs to identify patterns, predict future data points, and recognize trends. Using computer science concepts, students will write simple algorithms to model data changes and automate predictions. By applying mathematical principles, they will compare high and low values, determine frequency of occurrences, and draw conclusions from trends. This integration fosters both computational thinking and data analysis skills, preparing students for real-world problem-solving.

Science

- Students analyze simple tide tables and the phases of the moon over time to explain the relationship between the tides and the phases of the moon. Use the data from the table to predict which tide will occur on a particular moon phase.

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

4.DA.3 The student will create a computational model that represents attributes and behaviors associated with a concept.

- a. Examine models of physical objects and processes.
- b. Create a computational model that reflects the attributes and behaviors associated with a concept.
- c. Explain how a computer model illustrates a given concept.

Understanding the Standard

[4.DA.3a] Scientists, computer scientists, mathematicians, and programmers construct and use models to better conceptualize and understand phenomena under investigation or to develop a possible solution to a proposed problem. Models refer to computer created simulations, representations, diagrams, physical replicas, mathematical representations, and analogies that mimic the attributes and behaviors of a physical object or process.

Students use computer models to examine physical objects and processes and simulate real-world systems without needing to conduct experiments in the real world, which may not always be possible. Creating computational models allows students to visualize and interact with abstract ideas in a more concrete way, while using technology or software to simulate how these ideas work in real life. In earlier grades, the emphasis is recognizing that computers are a tool to construct models and identify the advantages and disadvantages of using a computer as a tool. The diagram below is a computer-generated model a student may create to represent the process of photosynthesis:

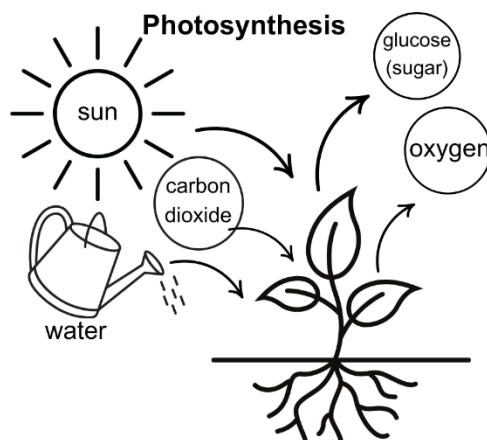


Image created with Canva

[4.DA.3b] Building on their knowledge of how to construct models, students examine various models of physical objects and processes so that they can create use a computer to create a computational model that reflects the attributes and behaviors associated with a concept.

This may involve writing code, using computer software, or designing an interactive system. Students extend their understanding of models by creating simple programs or animations to simulate processes such as how plants grow under different environmental conditions or variables (e.g., sunlight, water, temperature). Creating and manipulating these models allows students to predict how an independent variable (such as more sunlight or more water) can affect an outcome and understand and explain how real-world systems (like plant growth or weather patterns) behave. Consider the following example:

- A student designs a computational model using pseudocode to simulate the photosynthesis process in a simplified way, where different variables (such as light, water, and carbon dioxide) affect the output of glucose and oxygen.
- The model includes basic attributes such as sunlight (light intensity), water level, carbon dioxide concentration, glucose production, and oxygen production.
- The behaviors in the model reflect how the plant uses these resources to produce glucose and oxygen.
- The model also allows students to adjust variables and observe how changes in light, water, or carbon dioxide levels affect the rate of photosynthesis. The sample below represents some of the basic block coding input a student may use to model the process (not all pseudocode is present in the example).

Pseudocode example created by teacher after looking at online examples.

```
# Define plant attributes
# Define photosynthesis formula (simplified)
# Simulate the photosynthesis process over 5 cycles (e.g., days or hours)
for cycle in range(1, 6):
# Output the final results
```

By creating computational models, students develop the ability to interact with technology in ways that support inquiry and problem-solving across a range of subjects, from science and mathematics to economics and social studies.

[4.DA.3c] Students can demonstrate their understanding of how computer models are used to illustrate and explore concepts in computer science and related fields by explaining how their model illustrates a given concept. In the photosynthesis model above, a student may explain the concept in the following way:

- *“Photosynthesis is the process by which plants make their food using sunlight, water, and carbon dioxide. In the computer model, there is a plant with its roots taking in water and its leaves absorbing sunlight. The model shows how sunlight helps the plant turn carbon*

dioxide from the air and water from the soil into glucose (which the plant uses for energy) and oxygen (which the plant releases into the air). I can adjust variables in the model by adjusting the amount of light and water to see how these changes affect the photosynthesis process. For instance, when I add light, the plant produces more glucose, and when I reduce the amount of water, the plant produces less glucose. This helps me understand how each part of photosynthesis works together."

This type of explanation not only highlights the components and functionality of the computer model but also clearly connects the model's features to the content, helping to deepen the student's understanding of the concept.

Concepts and Connections

CONCEPTS

Models help represent physical objects and processes, allowing for analysis and simulation of real-world scenarios. Computational models use data and algorithms to reflect attributes and behaviors, enabling predictions and problem-solving. Computer models illustrate concepts by organizing information, demonstrating relationships, and simulating outcomes to enhance understanding and decision-making.

CONNECTIONS

Within this grade level: At this grade level, students expand their data analysis skills by using computational models to represent attributes and behaviors associated with a concept. They 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).

Vertical progression: In Grade 3, students created models of processes or physical objects and described how computers are used to design the models. They also explored the advantages and limitations of using a computer to design models (3.DA.3). In Grade 5, students will explore machine learning models to compare how training data is used, explain how it is used to make classification predictions, and discuss why large and diverse datasets are necessary in machine learning (5.DA.3).

ACROSS CONTENT AREAS

English

- **4.C.3A** The student will develop effective oral communication and collaboration skills to build a community of learners that process, understand, and interpret content together. A) Select, organize, and create engaging presentations that include multimedia components and visual displays.

Science

- **4.1** The student will demonstrate an understanding of scientific and engineering practices by a) asking questions and defining problems; b) planning and carrying out investigations; c) interpreting, analyzing, and evaluating data; e) developing and using models; f) obtaining, evaluating, and communicating information *4.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 4.1 include: 4.3 (organisms), 4.5 (planets), 4.6 (relationship among Earth, moon, and the sun), 4.7 (ocean environment), 4.8 (Virginia natural resources)*

Mathematics

- **4.CE.3** The student will estimate, represent, solve, and justify solutions to single-step problems, including those in context, using addition and subtraction of fractions (proper, improper, and mixed numbers with like denominators of 2, 3, 4, 5, 6, 8, 10, and 12), with and without models; and solve single-step contextual problems involving multiplication of a whole number (12 or less) and a unit fraction, with models.
- **4.MG.5a** The student will classify and describe quadrilaterals (parallelograms, rectangles, squares, rhombi, and/or trapezoids) using specific properties and attributes.
- **4.MG.6** The student will identify, describe, compare, and contrast plane and solid figures according to their characteristics (number of angles, vertices, edges, and the number and shape of faces), with and without models.
- **4.PS.2** The student will model and determine the probability of an outcome of a simple event.

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.

Science

- Students use a block-based coding program to design a computational model that demonstrates the difference between the rotation and the revolution of the Earth and the Sun and use it to explain how those motions affect day and night and seasons on Earth.
- Students construct a model to demonstrate the location and order of the planets in relation to the Sun and use the model to summarize our Solar System

Mathematics

- Students will explore multiplication by examining physical models of repeated groups, such as equal sets of objects. Then, they will use a block-based coding tool to create a computational model that visually represents multiplication by generating groups and calculating totals. Through coding, students will manipulate inputs to see how changes affect the output, reinforcing the behaviors and attributes of multiplication.

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

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

Impacts of Computing (IC)

4.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 Virginia's economy.
- b. Examine and explain how computing technologies influence and are influenced by culture.
- c. Identify social and ethical issues related to computing devices and networks.

Understanding the Standard

The use of technology, including computers, has allowed for global communication and has revolutionized the everyday access of information, whether for business, scientific or personal use. Although there are many positive impacts in using technology, there are also times when computer use has impacted us in undesirable ways.

As computer technology continues to advance and new generations of machines grow faster and have greater capabilities, computing technologies become more deeply fixed in daily life, magnifying both the benefits and the downside risks.

Positive Impacts	Negative Impacts
Easy access to information	Increase in sedentary lifestyles
Automated machinery	Family and leisure interruption
Fast and accurate data processing	Loss of privacy

[4.IC.1a] On the positive side, computing technologies have significantly boosted Virginia's economy through innovations in cloud computing, cybersecurity, and data analytics. Notable companies have established major data centers in the state, creating job opportunities.

It is equally important to consider the negative implications, such as the digital divide, where unequal access to these technologies can deepen social inequalities. Additionally, issues like privacy concerns, cybersecurity threats, and potential job displacement due to automation are critical challenges.

People can work in different places and at different times to collaborate and share ideas when they use technologies that reach across the globe. These social interactions affect how local and global groups interact with each other. As with any social interaction, there are acceptable behaviors that people should use when interacting with others.

[4.IC.1b] Computing technologies have significantly shaped culture by transforming how people communicate, access information, and entertain themselves. Social media platforms, for example, have created new ways for individuals to connect and share their lives, transcending geographical boundaries and fostering global communities.

[4.IC.1c] Streaming services and digital content have revolutionized media consumption, allowing for a more personalized and on-demand experience. Additionally, the rise of mobile apps and smart devices has made information and services more accessible, influencing daily habits and lifestyles.

Computing technologies continue to drive cultural changes by introducing new forms of interaction, creativity, and connectivity.

Concepts and Connections

CONCEPTS

Computing technologies impact the economic growth in Virginia, shaping numerous industries. As these computing technologies evolve, they influence societal norms, values, and expectations, often redefining community and participation in the digital age. Ethical concerns, such as data privacy and cybersecurity risks, highlight the need for responsible technology use.

CONNECTIONS

Within this grade level: At this grade level, Students are expanding on their knowledge of the impacts of computing. At this stage they will focus on the economic impacts on computing industries in Virginia (4.IC.1a).

Vertical progression: In Grade 3, students explored a global perspective on the impacts of computing and examined how computing technologies influenced culture along with social and ethical issues around the impacts of computing devices (3.IC.1). In Grade 5, students will predict consequences associated with inappropriate use of computing technologies and develop solutions to scenarios involving the inappropriate use of computing technologies (5.IC.1).

ACROSS CONTENT AREAS

History and Social Science

- **VS.9** The student will apply history and social science skills to understand the ways in which Virginia became interconnected and diverse by a) explaining the importance of railroads, waterways, new industries, and the growth of cities to Virginia's economic development in the late 1800s; and b) explaining the economic and social transition from a rural society to a more urban society.

- **VS.10** The student will apply history and social science skills to understand the role Virginians played in American history during World War I and World War II by a) examining how key leaders and citizens prepared for wartime; and b) describing the contributions made by military veterans and Medal of Honor recipients.
- **VS.13** The student will apply history and social science skills to explain Virginia's role in the global economy in the 21st century by a) examining major products and industries important to Virginia; and b) examining the impact of the ideas, innovations, and advancements of Virginians on a global market.

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.
- **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.

History and Social Science

- Students will explore how advances in communication technology have changed how people in Virginia share information, from newspapers and radios to the internet and social media. They will discuss the social and ethical issues that come with modern computing, such as online privacy, respectful communication, and misinformation.
- Students explore how advances in technology have changed life and work in Virginia over time. They will compare early technologies that require step-by-step instructions with modern tools that use machine learning to improve through patterns and examples. By examining how machine learning helps computers make decisions, students will connect this concept to how businesses, communication, and daily life.

Mathematics

- Students research and collect data on significant technological advancements and their economic contributions to Virginia. Students can research how technological advancements in our military and government have increased efficiency.

- Students apply the data cycle to analyze information about technological advancements in Virginia. Using data collected or provided (e.g., timelines, charts, or surveys), students will formulate questions, represent the data using appropriate graphical displays (such as bar graphs or line plots), and interpret the results. They will draw conclusions and make predictions about the potential future impact of computing on life and work in Virginia.

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

4.IC.2 The student will describe the impact of screen time on relationships at home and at school.

- a. Describe the impact of excessive screen time on maintaining friendships and family dynamics.
- b. Explain how playing video games and the use of social media can impact relationships and personal health.

Understanding the Standard

[4.IC.2a] Screen time is the time spent on a computing device. It has become an integral aspect of students' daily lives, deeply influencing their interactions and relationships both at home and in school. As technology becomes more accessible, its impact on social dynamics is increasingly profound.

At home, excessive screen time can diminish face-to-face interactions among family members, potentially leading to feelings of isolation or misunderstandings. Prolonged periods of playing video games or using social media can lead to reduced quality time with family and friends, impeding the development of critical social skills and emotional bonds. Furthermore, these activities can impact personal health, contributing to issues such as sleep disruption, reduced physical activity, and increased stress. It is important for students to understand the potential consequences of excessive screen use and to find a healthy balance that allows them to maintain meaningful relationships and enhance their overall well-being.

In school, overuse of screens may hinder students' ability to form meaningful peer relationships, sometimes resulting in social withdrawal and communication challenges. Nonetheless, balanced and mindful use of screen time also presents opportunities for learning, staying connected, and accessing support networks.

[4.IC.2b] Screen time has both positive and negative impacts, depending on how it is used. It can support learning, creativity, and connection with others, but excessive or unbalanced use may lead to reduced physical activity, sleep issues, or decreased social interaction. Teaching students to make mindful and purposeful choices about screen time is essential for developing healthy digital habits.

Positive Impacts	Negative Impacts
Enhanced Learning Opportunities	Reduced Direct Social Interactions
Interactive Educational Tools	Reduced Face to Face Interactions with Family Members
Access to Information	Difficulties Creating and Maintaining Peer Relationships
Skill Development	Less Participation in Family Activities
Staying Connected with Family and Friends	Reduction in Quality Family Time
Online Support Networks	Risk of Cyberbullying and Harassment
Access to Health Resources	Excessive Use and Sharing Leads to Privacy Risks

Strategies for Managing Screen Time:

1. **Set Clear Guidelines:** Establish clear rules about screen time duration and ensure a balance between online and offline activities. Encourage ‘tech-free’ times, such as during meals or family gatherings, to foster direct interactions.
2. **Encourage Quality Over Quantity:** Focus on the quality of screen time. Encourage activities that have educational value and promote positive interactions. Involve students in selecting educational content that aligns with their learning objectives.
3. **Promote Alternative Activities:** Encourage students to engage in physical activities, hobbies, and interests outside the digital world. Provide opportunities for social interaction through group activities, clubs, and extracurriculars.
4. **Foster Digital Citizenship:** Teach students about responsible and respectful online behavior, emphasizing the importance of digital etiquette and kindness. Discuss the impact of cyberbullying and the importance of reporting inappropriate online behavior.
5. **Integrate Screen Time with Learning Objectives:** Make use of educational technologies and platforms as part of the curriculum to enhance learning experiences. Use screen time as a tool to complement traditional teaching methods, ensuring it is purposeful and educational.

Balancing screen time is crucial for the holistic development of students. By fostering mindful use of technology, educators and parents can ensure that students harness the advantages of the digital age while mitigating its potential drawbacks.

Concepts and Connections

CONCEPTS

Excessive screen time can weaken relationships by reducing face-to-face interactions and emotional connections. While video games and social media can foster social engagement, overuse may lead to isolation, stress, and unhealthy habits. Balancing digital and offline interactions is key to maintaining strong personal connections and overall well-being.

CONNECTIONS

Within this grade level: At this grade level, students will develop plans for managing screen time. They will be able to explain the impact of excessive screen time on relationships and the impact at school (4.IC.2). Students will be able to build on prior knowledge to understand the impact of excessive screen time.

Vertical progression: In Grade 3, students gained an understanding of screen time and began to understand the importance of responsible screen time management (3.IC.2). In Grade 5, students will examine the impact of screen time on academic performance. They will compare and contrast screen time that benefits and hinders academic performance (5.IC.2).

Opportunities for Computer Science Integration

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

English

- Students will conduct research using multiple sources to examine the impact of social media on young children. They will evaluate the credibility and relevance of the information gathered, then plan and write a well-organized opinion piece stating whether or not fourth graders should have access to social media. Their writing will include a clear opinion, supporting reasons, and evidence from their research.

Mathematics

- Students will collect data on their personal screen time use over a set period of days, recording time spent on different types of activities (e.g., educational apps, gaming, texting, video streaming). They will organize the data into a table and represent it using an appropriate graph. Students will analyze the data to identify trends, compare categories, and draw conclusions based on their findings.

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

4.IC.3 The student will examine the impact of computing technologies in the workforce.

- a. Research and analyze the skills needed for careers in computing technology fields.

Understanding the Standard

[4.IC.3a] Computing technologies are transforming industries and redefining job functions across the global workforce. These technologies encompass a broad spectrum of tools and applications, including software development, artificial intelligence (AI), data analytics, cloud computing, cybersecurity, and robotics. Their far-reaching integration is evident in various sectors as these computing technologies are implemented to streamline operations, enable data-driven decisions, and enhance communication and collaboration. The following table gives suggestions on careers in the computing technology field that students can use for research.

Key Computing Technologies and Their Impact in the Work Force		
Key Technology	Impact	Example
Data Analytics	Businesses utilize data analytics to gather and analyze data, revealing patterns and insights that drive innovation, efficiency, and competitive advantage. For instance, retail companies use data analytics to manage inventory more effectively and understand consumer buying habits.	In fields such as e-commerce, companies use data analytics to personalize customer experiences, recommending products based on past purchases.
Cloud Computing	Cloud computing enables businesses to store and access data over the Internet, facilitating easy scalability and remote work.	By reducing the need for physical infrastructure, cloud services help companies save on costs associated with hardware and maintenance.
Cybersecurity	To ensure that data is secure from unauthorized access, businesses implement cybersecurity measures.	Regularly scanning for vulnerabilities in the platform and applying patches as soon as they are released helps protect against known threats.
Robotics	In manufacturing industries, robots are used to perform repetitive tasks with precision, like assembling parts on a production line. This not only increases efficiency but also enhances the safety of human workers.	Robotics is also making strides in healthcare through applications like robotic-assisted surgery, which allows for minimally invasive procedures with higher precision.

Educators should help students understand the broad impacts of these technologies. This includes discussing current trends and potential future advancements in their curriculum.

Educators should focus on developing both technical and soft skills. Technical skills might include coding, data analysis, and cybersecurity basics, while soft skills can encompass problem-solving, critical thinking, and adaptability.

Concepts and Connections

CONCEPTS

Computing careers require technical proficiency in areas like programming, cybersecurity, and data analysis, alongside analytical problem-solving skills. Staying informed on industry trends helps educators prepare students for evolving technology fields, ensuring they acquire the knowledge and adaptability needed for success.

CONNECTIONS

Within the grade level/course: At this grade level, students at this grade level will examine the impact of computing technologies on the workforce. They will examine the various key computing technologies and their impact on the workforce (4.IC.3).

Vertical Progression: In Grade 3, students researched careers related to computing technologies and their impact on society (3.IC.3). In Grade 5, students will examine the global impact of technology related careers along with examining opportunities of computing careers (5.IC.3).

ACROSS CONTENT AREAS

English

- **4.R** The student will conduct research and read a series of conceptually related texts on selected topics to build knowledge on grade-four content and texts, solve problems, and support cross-curricular learning.

History and Social Science

- **VS13.b** The student will apply history and social science skills to explain Virginia's role in the global economy in the 21st century by examining the impact of the ideas, innovations, and advancements of Virginians on a global market.

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.
C. Contribute constructively to project teams, assuming various roles and responsibilities to work effectively toward a common goal.

Opportunities for Integration

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

History and Social Science

- Students will explore careers in computing technologies in the state of Virginia.
- Students explore how federal agencies such as the FBI and CIA have contributed to the development and application of computing technologies for data analysis, encryption, surveillance, and national security purposes throughout history..

Mathematics

- Students will use the data cycle to collect data on the number of jobs in Virginia that involve computing technologies and compare it with the number of jobs in Virginia that do not involve computing technologies.

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. Building Relationships and Norms
2. Include Multiple Perspectives
4. Use Collaboration Tools

D. FOSTERING COMPUTATIONAL THINKING PRACTICES:

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

4.IC.4 The student will describe the importance of copyrights and intellectual property rights.

- a. Demonstrate an understanding of copyright and the fair use of information.
- b. Explain how intellectual property can be protected.
- c. Give proper attribution to the original author of digital and online content.

Understanding the Standard

[4.IC.4a] As students work with different artifacts (reference materials, resources, etc.), they should understand that these sources of information were created by others. Authors, illustrators, and programmers are responsible for the creation of many sources of information that are used in the classroom and at home. Fourth grade students need to avoid plagiarism by giving credit to sources appropriately.

[4.IC.4a] Copyright is a legal protection that gives creators of original works (like literature, art, music, software, etc.) exclusive rights to use and distribute their creations. It aims to recognize and reward the intellectual effort of creators.

[4.IC.4b] With computers, vast amounts of digital content are created and shared every day: software, games, digital art, music, videos, and more. Copyright ensures that the rights of creators are protected, preventing unauthorized use or distribution of their work. It encourages innovation and creativity by ensuring creators can benefit economically from their work, which motivates them to produce more high-quality content.

Intellectual Property Rights (IPR) are legal protections granted to creators and inventors for their original works, designs, and inventions. These rights help ensure that the creators can control how their creations are used and benefit from them.

Examples in Computing Technologies That are Protected from Unauthorized Use			
Software Programs	Digital Art and Graphics	Music and Videos	Websites and Content
Games and apps that developers create	Images that are created using digital tools	Songs and videos posted on the Internet	Text, images, and other content on websites

[4.IC.4c] Computing creates many opportunities, but it also raises ethical concerns. The ability to easily copy and share digital media, such as videos, photos, and music, can lead to unauthorized use, including online piracy and ignoring copyright rules, like failing to give credit to original creators. These actions not only violate intellectual property rights but also compromise digital integrity.

Related topics include plagiarism, fair use, and the importance of properly citing online sources. While students are not expected to know specific copyright laws at this level, they should understand the basic principles of respecting ownership and using content responsibly. Developing these habits supports ethical behavior in digital environments and encourages responsible content creation and sharing.

Concepts and Connections

CONCEPTS

Copyright and fair use play a vital role in digital citizenship, promoting ethical content use while safeguarding creators' rights. Intellectual property is safeguarded through legal measures like patents, trademarks, and copyrights. Proper attribution acknowledges original authors, reinforcing media literacy and fostering integrity in online interactions.

CONNECTIONS

Within the grade level/course: At this grade level, students at this level will describe the importance of intellectual property rights. They will be able to define intellectual property rights and understand that copyright gives legal protections. Students also need to know how to give proper attribution. They will also be able to define the various digital works that are protected against unauthorized use (4.IC.4).

Vertical Progression: In Grade 3, students explored how to use information created by others with permission. They demonstrated how to use information created by others (3.IC.4). In Grade 5, students will learn the difference between open-source licenses and copyright. They will also research the risks of inappropriate use of digital resources (5.IC.4).

ACROSS CONTENT AREAS

English

- **4.R** The student will conduct research and read a series of conceptually related texts on selected topics to build knowledge on grade-four content and texts, solve problems, and support cross-curricular learning.
F. Avoid plagiarism and give proper credit by providing citations whenever using another person's media, facts, ideas, graphics, music, and direct quotations.

History and Social Science

- **VS.h** The student will apply history and social science skills to the content by practicing civility, respect, hard work, honesty, trustworthiness, and responsible citizenship skills. citizenship skills. and texts, solve problems, and support cross-curricular learning.

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.
B. 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 examine various media messages (auditory, visual, and written) and give proper attribution to the original author of the media messages. They will analyze the purpose of each message, identify techniques used to influence the audience, and evaluate the effectiveness of those techniques. Students may practice summarizing the content in their own words while crediting the source appropriately to strengthen media literacy and responsible research habits.

History and Social Science

- Students will use secondary sources to analyze information; students can work on digital citizenship skills by showing respect to the original author and giving proper attribution.

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. Decompose 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)

4.NI.1 The student will identify the interrelationship between computing devices and a computing network.

- a. Define client and server.
- b. Describe how packets are used to transmit information on a network.
- c. Describe factors that may affect the speed of data transmission.
- d. Differentiate between networking tasks that require Internet access and tasks that do not require Internet access.
- e. Model how computing devices in a network transmit and receive information.

Understanding the Standard

Computing devices can be standalone or connected to a computing network. Computing devices include things like laptops, desktops, tablets, and smart phones. All of these devices can work alone, but when connected to a network, they become even stronger.

Computers that are part of a network transmit information through both physical and wireless pathways which are known as a network. Network pathways allow communication between computers within the same building or different locations around the world. These communications exist in a variety of forms to include emails, blogs, images, videos, social media platforms, and online video games.

Computer networks allow collaboration and sharing of information as well as resources like storage space and printers. Networks can store large amounts of information that multiple devices can access as it is needed. When services like Netflix or Disney+ are used, videos that are not stored on the personal device can be watched. In schools, printers and copiers are often located in central locations that can be shared by many individuals. Often on a computer network, the parts may be referred to as a client and a server. In this relationship, they work together to accomplish a task much like a server in a restaurant delivers the food to the client or customer.

[4.NI.1a] The client-server relationship is a way computers communicate over a network. The client makes a request, and the server responds, enabling access to shared resources or services. This relationship allows devices like computers or browsers (clients) to access resources such as websites, files, or databases hosted by servers.

- Client is computer or software application that requests information from another computer or server over a network.
- Server is a powerful computer that sends information back to the client. It stores information and sends it to other devices, called clients, when they ask for it.

[4.NI.1b] When information is sent over a network, like the Internet, the computer sending the information breaks it down into packets. Packets are small pieces of information. The packets travel to their destination using routers that direct them where to go. Each device has a unique

address. If one device wants to send information to another, it writes the information in small packets. Each packet has a "To" and "From" address, so it knows where it's going. When the packets reach their destination (another computer), they are put back together in the right order to show the whole picture. This process allows the sender and the receiver to see the same text, image, video, etc.

[4.NI.1c] Several factors can affect how fast data travels:

- **Connection Strength:** If you're connected to strong Internet, like when you're close to a Wi-Fi router, data moves faster. But if the connection is weak, it can be slow.
- **Amount of Data:** Big files, like videos, take longer to send than small files, like short messages.
- **Network Traffic:** If lots of people are online and using the same network, things slow down, just like traffic on a busy road.
- **Distance:** Data has to travel from one place to another. The farther it has to go, the longer it can take.
- **Type of Connection:** Different types of Internet connections have different speeds. Fiber connections are very fast, while older types of connections can be slower.

[4.NI.1d] Some networking tasks require the use of the Internet, while others do not. See the table for some examples.

Does it need the Internet?	
Requires Internet	Does NOT Require Internet
• Watching YouTube videos or streaming shows needs the Internet because the video is stored on computers far away. • Playing online games with people in other places requires the Internet to connect to game servers. • Sending an email or using social media also needs the Internet to reach people all over.	• Printing a document from your computer to a printer nearby doesn't need the Internet if they're on the same local network. • Sharing files between computers or tablets in the same room, like in a school lab, doesn't need Internet—just a local network. • Playing games together on local devices (like tablets connected directly) doesn't need the Internet.

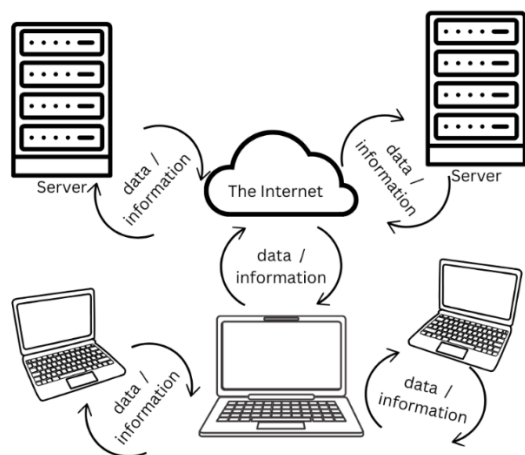


Image created with Canva

Concepts and Connections

CONCEPTS

Computing devices communicate by transmitting data in small units called packets, which travel across networks to deliver information efficiently. Clients request services, while servers process and respond to those requests. Factors like bandwidth, network congestion, and latency affect transmission speed. Some tasks, such as browsing the web require Internet access, while others, like local file sharing, do not. Understanding how devices exchange data resources ensure reliable connectivity and optimized performance.

CONNECTIONS

Within the grade level/course: At this grade level, students continue to explore how devices are connected and transmit information. Students may use computational thinking skills to simplify concepts and look at patterns in how information travels from one device to another and what factors may impact the speed of the exchange.

Vertical Progression: In Grade 3, students differentiated between a network and the Internet and identified the components of a network. Students began to identify ways networks transmit information (3.NI.1). In Grade 5, students explain the factors that impact the speed data transmission can impact cloud computing which is covered in grade five (5.NI.1).

ACROSS CONTENT AREAS

History and Social Science:

- **VS.13a** The student will apply history and social science skills to explain Virginia’s role in the global economy in the 21st century by examining major products and industries important to Virginia
- **VS.13b** The student will apply history and social science skills to explain Virginia’s role in the global economy in the 21st century by examining the impact of the ideas, innovations, and advancements of Virginians on a global market.

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 the exchange of data has changed over time from the oral stories of Native Americans to the digital communication of today. Discuss positive and negative impacts to society due to the change in how information is transferred.

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

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 enhance 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 the 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 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 4 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.
Artificial Intelligence (AI)	A domain of computer science that focuses on the research and development of computers and systems that simulate or replicate tasks that require human intelligence. [This is not formally introduced to students until Grade 6]
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.
American Standard Code for Information Interchange (ASCII)	A character encoding standard that represents text in numeric form, enabling multiple data representations in computing devices.
Acceptable Use Policy (AUP)	Rules and guidelines that define safe practices and responsible use of technology.
Assigning	Setting a value to a variable either manually or based on program logic.
Attribution	Giving credit for work created by someone else.
Author	The creator of a book, image, song, or object.
Binary	A number system that uses two digits, 0 and 1.
Bit	The smallest unit of data in computing.
Block-Based Programming	A visual drag and drop programming tool that users can use to create programs using command blocks.
Boolean	A data type that can only have two possible values: true or false; essentially representing a logical state where something is either true or not.
Byte	A unit of digital information that's made up of eight binary digits, or bits.
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.
Cleaning	Making sure that the data is correct, consistent, and clear.
Client	A computer or software application that retrieves and uses information, resources, or services from another device over a network.
Cloud Computing	Storing and accessing information on the Internet
Code	Any set of instructions expressed in a programming language.
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.
Digital Divide	The difference in access to technology and the Internet between people who have access and those who do not.
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.
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.
Expression	In a programming language, a combination of explicit values, constants, variables, operators, and functions interpreted according to the particular rules of precedence and of association which computes and then produces (returns, in a stateful environment) another value.
Extraction	Process of identifying, isolating, or deriving meaningful information.
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.
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.
Initialize	Defining a variable and assign it an initial value.
Intellectual Property	A person's own creations of the mind, such as inventions, drawings, stories, and poems.
Intellectual Property Rights (IPR)	Legal protections granted to creators and inventors for their original works, designs, and inventions.
Internet	A global network of interconnected computing devices that allows devices to share information and resources.
Internet Protocol (IP) Address	A unique numeric value assigned to a computer or other device connected to the Internet so that it may be identified and located.
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.
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.
Machine Learning	A process that occurs when computers learn from examples instead of following exact instructions. Examples of machine learning include supervised learning, unsupervised learning, and reinforcement learning.

Memory	The physical storage in computing devices where data is processed and instructions for processing are stored. Memory types include RAM (Random Access Memory), ROM (Read-Only Memory), and secondary storage like hard drives, removable drives, and cloud storage.
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.
Modifying	Changing the value of a variable as the program runs, often in response to user input or program events.
Naming Convention	Guidelines for the use of descriptive names for variables, functions, and classes.
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.
Open-source License	People can use, change, and share a computer program (like an app or a game) for free, as long as they follow some simple rules.
Output	Data or information produced by a computing device after processing input.
Output Devices	Hardware components that display processed data or information.
Packet	Packets are smaller pieces of data that can be sent across networks.
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.
Phishing	Deceptive online attack where scammers pretend to be a trusted source and trick individuals to share personal identifiable information.
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.
Purposeful Use of Computing Devices	Includes the understanding that technology has specific purposes, such as learning new things, solving problems, and communicating.
Reboot	To turn off the device and turn it back on.
Reinforcement Learning	A type of machine learning where a computer learns through trial and error. The computer receives feedback and adjusts its behavior to improve performance.
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.
Server	A computer or software application that provides information, resources, or services to clients over a network.
Simulation	Replicating the behavior of a real-world process or system over a period of time.
Social 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).
String	A string is a series of letters, numbers, or symbols. It can represent things like a name, address, or song title. Common operations with strings include finding their length, joining them together, and extracting parts of them.
Supervised Learning	A method of machine learning that involves a computer using large amounts of labeled datasets to recognize patterns, classify data, and make predictions.
Switch	Devices that act as a bridge within a network and connect multiple devices (like computers, printers, and routers) on a LAN (local area network) network allowing them to send and receive data to communicate directly with each other.
Syntax	Rules or structure of a programming language.
Syntax Errors	An error caused by a mistake in the rules or structure of a programming language, such as missing parentheses, commas, or incorrect keywords.
Table	A structured format to organize and record information in rows and columns.
Testing	The process of evaluating a program or system to assess its results and outputs, ensuring accuracy, performance, and reliability, while identifying errors.
Texting/text Message	A short, written message sent from one phone to another over the Internet or a mobile network.
Training Data	A collection of labeled or unlabeled data used to teach a machine learning model how to recognize patterns, make predictions, or perform tasks.
Trends	Are long-term directions or movements in data or behavior that indicate a general tendency or shift over time.
Troubleshoot	Process used to diagnose why a system or process is not working as expected and systematically testing solutions to resolve the issue.
Two-way Branching Conditional Control Structures	A programming concept that allows a program to make a decision based on two different paths.
Unauthorized Access	Information is accessed without the permission of the owner.

Unsupervised Learning	When a computer learns to find groups or patterns without anyone telling it what's right or wrong.
User Input	Data or information that a user provides to a computer program during its execution (2024). Data that is taken in by a computer for processing (2017).
Username	A unique name that people use to log into a device or online account. It is like a nickname that helps the computer recognize who is logging in.
Variables	A programming element that is a named storage location in memory that holds a value, which can be modified during the execution of a program.
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.