2024
Computer Science Standards of Learning

Grade 2 Instructional Guide

VIRGINIA DEPARTMENT OF EDUCATION

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

Superintendent of Public Instruction
Emily Anne Gullickson

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

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

Table of Contents

2024 COMPUTER SCIENCE *STANDARDS OF LEARNING*

Introduction

Virginia's Computer Science *Standards 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.

Second Grade: 2024 Computer Science *Standards of Learning*

In Second Grade, students apply computational thinking skills as they identify patterns and create algorithms for comparing objects based on attributes. Students follow the iterative design process, planning, implementing, and evaluating algorithms with events and loops using a block-based programming language. Proficiency is demonstrated in building, testing, and debugging programs. The use of computational thinking is expanded beyond programing and applied through abstraction and manipulation of data to create models and representations. Students define and categorize input and output components, explaining how computing systems acquire input and produce output. Students utilize the Internet to gather information and for collaborative tasks, while demonstrating safe behaviors when interacting with others and using the information. While using online resources, students learn the importance of password usage for privacy.

Algorithms and Programming (AP)

2.AP.1 The student will apply computational thinking to identify patterns, and design algorithms to compare and contrast objects based on attributes.

a. Compare and contrast multiple ways to sort a set of objects.
b. Create a table of features to organize objects.
c. Design an algorithm to sort objects into categories based on multiple attributes.

2.AP.2 The student will plan and implement algorithms that consists of events and loops using a block-based programming language.

a. Plan and create a design document to guide the construction of a program using plain language or pseudocode.
b. Identify a section of repeated actions within an algorithm and replace it with a loop.
c. Construct step-by-step instructions that include events and repetition.

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

a. Define program.
b. Read and interpret a program expressed in a block-based programming language.
c. Analyze and describe the results of a program.
d. Create and test a program that uses events and loops.
e. Revise and improve a program to produce desired outcomes.

Computing Systems (CSY)

2.CSY.1 The student will describe the characteristics of computing systems including hardware, software, input, and output.

a. Describe how hardware and software work together to accomplish a task.
b. Define and categorize components as inputs and outputs.
c. Describe how a computing system receives input and provides output.
d. Discuss how computers use binary code to communicate and process information.

2.CSY.2 The student will demonstrate an understanding of how to troubleshoot simple hardware and software problems that may occur during use.

a. Propose solutions to simple hardware and software issues.
b. Use appropriate steps to perform simple troubleshooting tasks.

Cybersecurity (CYB)

2.CYB.1 The student will model safe and responsible behaviors when using information and computing technologies.

a. Explain the need for safe and responsible uses of computing technologies.
b. Create a flowchart to illustrate the process for reporting inappropriate use of technology at school or at home.
c. Demonstrate and model safe and responsible behaviors when using computing technologies and online communication.

2.CYB.2 The student will explain the importance of using passwords to protect private information.

a. Identify and classify passwords as strong or weak.
b. Explain how a strong password helps protect the privacy of information.
c. Explain the risk of sharing passwords.

Data and Analysis (DA)

2.DA.1 The student will analyze data to make decisions with or without a computing device.

a. Collect and record numeric and non-numeric data and describe possible patterns.

b. Create questions that can and cannot be answered by the data.
c. Analyze data to draw conclusions and make decisions.

2.DA.2 The student will manipulate data, create representations, and evaluate data to solve a problem.

a. Create charts, graphs, and models using abstraction to represent data.
b. Analyze data visualizations to draw conclusions.
c. Propose and evaluate a solution to a problem or question based on data and/or data visualization.

Impacts of Computing (IC)

2.IC.1 The student will examine the positive and negative impacts of how using computing technologies has changed the way people live, work, and interact.

a. Identify current uses of computing/emerging technologies and discuss how they impact society.
b. Compare and contrast appropriate and inappropriate online behaviors that apply in the physical environment and the online environment.
c. Model healthy habits for using computing technologies.

2.IC.2 The student will explain the need to balance screen time and other activities.

a. Discuss appropriate times and places for screen use.
b. List and describe alternatives to screen time.

2.IC.3 The student will explain how computing technologies have an impact on the workforce.

a. Explain how computing technology is used in various careers.
b. Identify skills needed for careers that use computing technologies.
c. Discuss how computing technologies have changed the workplace.

Networks and the Internet (NI)

2.NI.1 The student will demonstrate the use of the Internet in gathering information to accomplish a task.

a. Explore ways information is organized and shared on the Internet.
b. Gather information from the Internet.
c. Summarize collected information using own words.

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 2: Computer Science Instructional Guide

In Second Grade, students apply computational thinking skills as they identify patterns and create algorithms for comparing objects based on attributes. Students follow the iterative design process, planning, implementing, and evaluating algorithms with events and loops using a block-based programming language. Proficiency is demonstrated in building, testing, and debugging programs. The use of computational thinking is expanded beyond programing and applied through abstraction and manipulation of data to create models and representations. Students define and categorize input and output components, explaining how computing systems acquire input and produce output. Students utilize the Internet to gather information and for collaborative tasks, while demonstrating safe behaviors when interacting with others and using the information. While using online resources, students learn the importance of password usage for privacy.

Algorithms and Programming (AP)

2.AP.1 The student will apply computational thinking to identify patterns and design algorithms to compare and contrast objects based on attributes.

a. Compare and contrast multiple ways to sort a set of objects.
b. Create a table of features to organize objects.
c. Design an algorithm to sort objects into categories based on multiple attributes.

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.

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

- Decomposition is the process of breaking apart a problem, process, or task into smaller, more manageable components. This involves identifying and recognizing relationships among the parts.
- Pattern recognition involves identifying commonalities, similarities, or differences in recurring elements.
- Abstraction is a filtering process. It enables one to focus on important and relevant information, while excluding or hiding irrelevant or less important details.

- Algorithmic thinking is the process of developing algorithms logical, systematic, and procedural ways to solve problems or finish tasks.

Computational Thinking: Example	
Decomposition	You are given a line of toy cars that need counting. You want to think of a way to make counting the cars easier.
Pattern recognition	You recognize the cars are arranged in groups of two.
Abstraction	You ignore the colors of the cars and the size of the cars.
Algorithmic thinking	You start counting the cars and recording two more each time you count.

[2.AP.1a] Objects can be observed and described based on their attributes; these attributes allow people to group items. An attribute refers to a characteristic or quality that describes and differentiates objects or data. Attributes of objects may include properties such as color, size, shape, weight, position, number, or texture. At this grade level, students will identify attributes in a set of objects and sort objects by predetermining sorting methods. Sorting is a process that involves comparing a set of objects to identify similarities and differences. This process strengthens logical thinking and helps students observe recurring structures, which is essential for pattern recognition.

Objects can be sorted differently based on chosen attributes. At this grade level, students will compare and contrast multiple ways to sort. Students will communicate methods used for sorting and explain the commonalities and differences in a set of objects. This strengthens students' ability to recognize patterns and make predictions based on patterns. In block-based programming environments, commands are grouped into categories based on function. In higher level programming languages, data is often classified by the type and format of the information.

[2.AP.1b] A feature table is a tool that helps organize information in a structured way. The feature table can be used to compare and classify objects based on attributes or characteristics. Using a feature table hones students' ability to recognize patterns and ability to organize and analyze data.

- Example of a feature table. This example and some of the features of the listed computing devices may vary depending on make and model of the computing device.

Computing Device	**Has a screen**	**Connects to the Internet**
Programmable Robot	Yes	No
Laptop	Yes	Yes

[2.AP.1c] The algorithm design process is a step-by-step way of creating instructions that help solve problems or complete tasks efficiently. Algorithms are clear, ordered sets of directions that a computer or person can follow. Good algorithm design means choosing the best steps, making sure they are organized well, and testing them to be sure they work correctly and quickly. Algorithms are finite set of step-by-step instructions designed to solve a problem or perform a task.

Concepts and Connections

Concepts

Computational thinking can be applied to identify patterns and design algorithms that compare and contrast objects based on their attributes. This process involves analyzing the characteristics of objects, recognizing similarities and differences, and using logical reasoning to create systematic steps for comparison.

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 patterns, and design algorithms to compare and contrast objects based on attributes.

Vertical Progression: In Grade 1, students used decomposition, pattern analysis, and algorithmic thinking to sort items into categories based on multiple attributes and create patterns (1.AP.1). In Grade 3, students expand algorithmic thinking to include extending patterns, processes, or components of a problem (3.AP.1).

Across Content Areas

Mathematics

- **2.MG.4** The student will describe, name, compare, and contrast plane and solid figures (circles/spheres, squares/cubes, and rectangles/rectangular prisms).

- **2.PFA.1** The student will describe, extend, create, and transfer repeating and increasing patterns (limited to addition of whole numbers) using various representations.
  c) Create a repeating or increasing pattern using various representations (e.g., objects, pictures, numbers).

Science

- **2.1** The student will demonstrate an understanding of scientific and engineering practices by planning and carrying out investigations. *2.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 2.1 include: 2.3 (matter), 2.4 (plants and animals), and 2.6 (weather).*

DIGITAL LEARNING INTEGRATION

- **K-2.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  C. Break problems into component parts, extract key information, and develop descriptive models, using technologies when appropriate, to understand complex systems or facilitate problem-solving.

Opportunities for Integration

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

English

- Students listen to or read "The Three Little Pigs" and discuss the main idea and key details; identify patterns in the story, such as the wolf's repeated attempts to blow down the houses and the different materials used by the pigs; and break down the story into elements, recognize similarities and differences between the pigs and their houses.
- Students design algorithms to compare these attributes using a Venn diagram.

Mathematics

- Students create and extend repeating patterns (e.g., red, blue, red, blue).
- Students create increasing patterns (e.g., one block, two blocks, three blocks) using blocks.
- Students break down and analyze the patterns to identify similarities and differences.

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

2.AP.2 The student will plan and implement algorithms that consists of events and loops using a block-based programming language.

a. Plan and create a design document to guide the construction of a program using plain language or pseudocode.
b. Identify a section of repeated actions within an algorithm and replace it with a loop.
c. Construct step-by-step instructions that include events and repetition.

Understanding the Standard

At school and at home, students engage in step-by-step activities on a routine basis. These may include such activities as brushing their teeth or preparing to leave school at the end of the school day. When students document these step-by-step instructions, they create algorithms. Algorithms are step-by-step instructions designed to solve a problem or perform a task. Algorithms are finite and sequential, with each step following a specified order. Algorithms can be created with or without computing devices.

In Grade 2, students begin to develop programs using block-based programming environments. A block-based programming language environment is a visual coding platform designed to help learners, especially elementary students, understand programming concepts without needing to write complex text-based code. Instead of typing instructions, users create programs by dragging and connecting graphical blocks. Each block has a specific function (e.g., move, turn, repeat) and can relate to other blocks to form a sequence, or algorithm, that tells the computer or app what to do.

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

Design documents are used in the planning process to serve as a bridge between abstract thinking and concrete solutions, reinforcing computational thinking. These documents often use plain language, pseudocode, or flowcharts to represent the algorithmic design and illustrate the programming process. Through the use of plain language or pseudocode students can focus on the problem-solving process, without the need to understand programming structures or syntax. Plain language refers to simple, everyday language used in speaking and writing. Pseudocode combines plain language and programming keywords to outline steps in an algorithm. Flowcharts are diagrams that show a sequence of steps to complete a task or solve a problem. Flowcharts use arrows and symbols to show the process or flow of actions in order

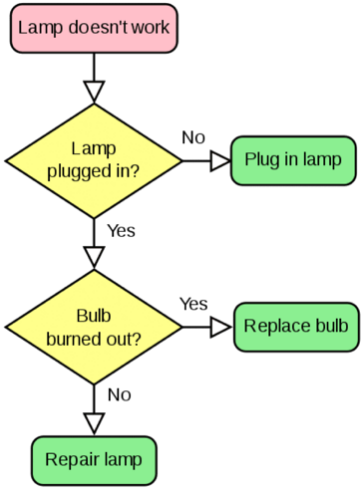


Image source: Flowcharting in Business Project Management

[2.AP.2b] In computer science, a sequence refers to a specific order in which instructions or steps are executed in an algorithm or program. Some instructions within a sequence may contain repeated actions. These repeated actions can be replaced with a loop. A loop is a set of instructions that are repeated until a specified condition is met, or predetermined number of repetitions has occurred.

Loops are a fundamental programming concept used to repeat a set of instructions multiple times until a specific condition is met or for a predetermined number of cycles. They allow for efficiency by reducing redundancy and automating repetitive tasks. A loop is a set of actions that is repeated until a certain condition is met. By recognizing repeating patterns in an algorithm, loops can replace repetitive steps, keeping the program structured and easy to modify. Applying an understanding of events and loops helps modify and enhance a program, ensuring it operates efficiently

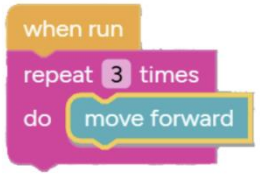
Understanding loops involves recognizing patterns, including repeating and growing sequences. In a repeating pattern, each unit follows a consistent cycle, maintaining the same structure throughout. A growing pattern, however, changes with each repetition by adding a new element, creating a progressive sequence rather than simple repetition. Recognizing these patterns helps in building structured algorithms that respond dynamically to changes and simplify complex tasks in programming.

Repeating and growing patterns play a fundamental role in both mathematics and computer science. In mathematics, they support the development of algebraic reasoning by helping learners recognize sequences, relationships, and functional rules. In computer science, these patterns contribute to computational thinking, as they form the basis for creating algorithms and understanding loops.

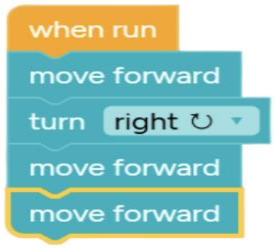
Sample numeric patterns include:

- 6, 9, 12, 15, 18,....(growing pattern);
- 1, 2, 4, 7, 11, 16,….(growing pattern);
- 20, 18, 16, 14,…..(growing pattern); and
- 1, 3, 5, 1, 3, 5, 1, 3, 5,.....(repeating pattern).

Loop saves times for the developer, improves code readability, and increases program efficiency.

Code block of a sequential algorithm where a task (move forward) is repeated three times. The action could be repeated depending on the task being performed.  Image source: Code.org	Code block of a sequential algorithm converted to a loop. The loop causes the task (move forward) to be repeated three times. The number in the loop can be changed to the number of actions being performed for a given task.  Image source: Code.orgode.org

[2.AP.2c] An event is something that causes all of a program or only a certain portion of the program to run (e.g., mouse clicks on the run block). Students also learn they can create events in their algorithm to signal to the computer that it needs to perform a certain action.



In a block-based programming language, the **"when run"** event is represented by a colored code block, typically orange. When a student clicks **"run,"** the code executes in sequence according to the instructions within the block. Code block example. Image source: Code.org

Concepts and Connections

CONCEPTS

Algorithmic thinking can be applied to design and construct programs that respond to events and include repetition through loops. Loops are used to repeat actions and events trigger responses when specific conditions are met. This process involves planning with plain language or pseudocode, identifying repeated actions within an algorithm, and organizing step-by-step instructions that structure the program efficiently and logically.

CONNECTIONS

Within the grade level/course: At this grade level, students continue building upon their algorithm design by planning and implementing algorithms that consist of events and loops using a block-based programming language (2.AP.2)

Vertical Progression: In Grade 1, students planned and implemented algorithms that included the use of sequence, and an event based on a predetermined task (1.AP.2). In Grade 3 students build upon their algorithmic design by including conditional control structures (3.AP.2).

ACROSS CONTENT ALIGNMENT

History and Social Science

- **2.10** The student will describe the contributions and roles of changemakers in United States history including but not limited to a) Scholars and Inventors: Benjamin Franklin, Benjamin Banneker, Thomas Jefferson, George Washington Carver, Booker T. Washington, Orville and Wilbur Wright, Steve Jobs, Jonas Salk, Thomas Edison, Alexander Graham Bell, and Mary Jackson.

Science

- **2.1** The student will demonstrate an understanding of scientific and engineering practices by planning and carrying out investigations.
  2.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 2.1 include: 2.4ab (animal life cycle and plant life cycle)

DIGITAL LEARNING INTEGRATION

- **K-2.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.
- **K-2.ID** Students use a variety of technologies, including assistive technologies, within a design process to identify and solve problems by creating new, useful or imaginative solutions or iterations.
  B. Select and use appropriate technologies to plan and manage a design process that considers design constraints and calculated risks.
  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.

History and Social Science

- Students illustrate Rosa Parks' bus story in sequence.
  - Students create a diagram for the events (example, a speech bubble for the driver - starting event) .
  - Students use a movable paper "bus" to represent the bus route .
  - Students relate this to their regular bus route (loop).
  - Students connect history with the algorithmic concepts of sequencing, events, and repetition.

Science

- Students explain how flowers grow from seeds, explain that students will make a paper flower to show its growth.

- Students cut out and color a seed, a small sprout, a bigger sprout with leaves, and a full-grown flower.
- Students connect life cycles and algorithms by planning the sequence: seed (event 1), small sprout (event 2) bigger sprout (event 3) full-grown flower with seed head (event 4), when the seed drops the pattern repeats (loop).
- Students make the connection that the plant life cycle is the loop that is repeated in nature.

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

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

a. Define program.
b. Read and interpret a program expressed in a block-based programming language.
c. Analyze and describe the results of a program.
d. Create and test a program that uses events and loops.
e. Revise and improve a program to produce desired outcomes.

Understanding the Standard

Programmers evaluate and refine their code to maintain program quality and ensure functionality The iterative process is a foundational skill that should be introduced early and reinforced across disciplines. It involves systematically developing, evaluating, and refining algorithms to ensure accuracy, efficiency, and effectiveness. This process typically includes designing an initial version, testing its functionality, collecting feedback, and making ongoing improvements to optimize the outcome. Through iteration, students develop critical problem-solving skills, foster creativity, and build resilience by learning from experience.

Iteration serves as a key method for validating whether an algorithm accurately represents the steps needed to complete a task. This approach is applicable in both computing environments and environments without computing devices, allowing learners to apply computational thinking across various settings. By engaging in iterative refinement, individuals strengthen their ability to analyze, test, and revise logical sequences, reinforcing the connection between process and outcome.

The iterative process reinforces a systematic approach to debugging and optimization, where errors are analyzed as diagnostic feedback to improve algorithmic outcomes. When an algorithm does not perform as expected, students must examine its logic to identify and implement necessary modifications, such as adding, removing, reordering, or altering steps, to achieve the intended result. Through repeated testing and refinement, students develop proficiency in recognizing patterns, identifying logical inconsistencies, and applying corrective strategies. The continuous cycle not only enhances computational thinking but also deepens students understanding of problem-solving and design thinking, strengthening both analytical skills and algorithmic fluency.

[2.AP.3a] A program is a complete set of instructions written in a programming language that a computer executes to perform a specific task. Code refers to the individual commands or statements that, when combined, form a program. Programming is the process of designing, writing, testing, and refining code to develop functional software that performs intended tasks.

[2.AP.3b] Tracing is a fundamental skill that involves systematically following each step of an algorithm to understand its behavior and predict its outcomes prior to implementation. By analyzing the sequence of operations along with inputs and expected outputs, tracing enables students to interpret and predict how a program will execute. It plays a crucial role in the design and development of algorithms and programs, ensuring accuracy and efficiency before implementation

 Image source: Code.org	Tracing Steps for the Event: 1. Event Trigger: The program begins when the "when run" button is clicked. 2. Move Forward - Step 1: The character moves forward once. 3. Move Forward - Step 2: The character moves forward a second time. 4. Move Forward - Step 3: The character moves forward a third time. 5. Turn Left: After completing the three forward moves, the character turns left.

[2.AP.3c] When a program runs, analyzing its behavior step by step is essential to understanding the code's structure and logic. Tracing the sequence of instructions and the flow of data reveals how the program processes information and how its components interact. The understanding of execution flow and detailed analysis allows student the ability to identifying logical relationships, explain program behavior, and making necessary improvements.

[2.AP.3d] Design documents are detailed plans that outline the structure, features, and implementation strategy of a project. It serves as a blueprint, providing clear specifications, goals, and guidelines for developers, designers, and stakeholders. Design documents often include diagrams, technical requirements, workflows, and rationale to ensure a shared understanding of the project's direction and execution. In programming, design documents include plain language, pseudocode, or simple diagrams. Plain language refers to simple, everyday language used in speaking and writing. In contrast, pseudocode combines plain language and programming keywords to outline steps in an algorithm. Flowcharts are diagrams that show a sequence of steps to complete a task or solve a problem. Flowcharts use arrows and symbols to show the process or flow of actions in order. Design documents serve as a bridge between abstract thinking and concreate solutions, reinforcing computational thinking.

A loop is a set of actions that is repeated until a certain condition is met. In a block-based programming language, designing a program using events and loops helps create efficient and organized code. Loops are essential programming constructs that allow a set of instructions to repeat until a specific condition is met or for a defined number of times, reducing redundancy and improving efficiency. By recognizing repeating patterns in an algorithm, loops can replace repetitive steps, keeping the program structured and easy to modify.

Events are a fundamental concept in programming that enables programs to respond to user interactions or system triggers. They act as signals that initiate specific actions within a program, allowing dynamic and interactive behavior.

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

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

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

 **Code Block A** Image source: Code.org	When the 'when run' button is clicked, the character first turns right, then moves forward five times.
 **Code Block B** Image source: Code.org	The improved version incorporates a loop. When executed, the character first turns right, then moves forward continuously until reaching the goal. The loop ensures a concise program by eliminating redundancy and unnecessary code. While both algorithms function correctly, the second version is more efficient.

Concepts and Connections

Concepts

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

Connections

Within the grade level/course: At this grade level, students build their use of the iterative design process to create, test, and debug a program containing events and loops in a block-based programming tool (2.AP.3).

Vertical Progression: In Grade 1, students used the iterative design process to construct, test, and debug algorithms that included sequencing and an event (1.AP.3). The algorithms may have been created during an unplugged activity where no computing device was used or through a plugged activity where a computing device and block-based programming tool was used. Grade 3 students expand their use of the iterative design process to create, test, and debug programs containing events, loops, and conditional structures in a block-based programming tool (3.AP.3).

Across Content Alignment

Mathematics

- **2.PFA.1** The student will describe, extend, create, and transfer repeating and increasing patterns (limited to addition of whole numbers) using various representations.

Digital Learning Integration

- **K-2.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  C. Break problems into component parts, extract key information, and develop descriptive models, using technologies when appropriate, to understand complex systems or facilitate problem-solving.
- **K-2.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.
- **K-2.EL** Students leverage technologies, including assistive technologies, to take an active role in choosing, achieving, and demonstrating competency in their learning goals, informed by the learning sciences.
  C. Use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.

Opportunities for Integration

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

English

- Students use the iterative design process to plan, create, test, and refine an interactive story using a block-based programming tool. Starting with a simple program featuring a character and basic movement, they add events and dialogue to enhance interactivity and narrative flow. Throughout, students apply sequencing, cause-and-effect reasoning, and audience awareness, then present their final stories and reflect on their design choices and challenges.

Mathematics

- Students create pictures using only symmetrical shapes, following a structured design process. Students then engage in an iterative design process by posing guiding questions, making targeted refinements through debugging, and revising their work for continuous improvement. This cycle continues until they produce a version that meets their intended goals. During a culminating "show and tell," students present their artifact and reflect on how the iterative process influenced their design choices and supported their problem-solving and creativity.

Skills in Practice

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

B. Fostering Iterative Design Practices:

1. Decompose 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)

2.CSY.1 The student will describe the characteristics of computing systems including hardware, software, input, and output.

a. Describe how hardware and software work together to accomplish a task.
b. Define and categorize components as inputs and outputs.
c. Describe how a computing system receives input and provides output.
d. Discuss how computers use binary code to communicate and process information.

Understanding the Standard

Computing systems are composed of different components. These components enable the user to complete different tasks using a computing system. A computing system is a group of hardware and software that work together to complete various tasks. Understanding how these parts interact helps users make better use of technology and troubleshoot problems when they arise.

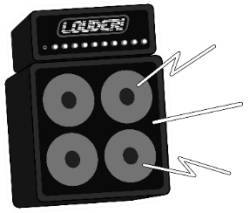
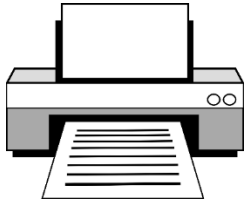
[2.CSY.1a] A computing system is an electronic device designed to receive, process, store, and produce or send information. It consists of hardware and software that work together to perform tasks. Hardware refers to the physical components of a computing system that can be seen and touched. This includes a monitor or screen for displaying information, buttons to power the device on and off, as well as a mouse and keyboard for inputting data. Hardware also encompasses the microprocessor or central processing unit (CPU), the system's command center, which processes all incoming (input) and outgoing (output) data.

Software is a collection of programs or instructions that direct the hardware on what tasks to perform. Unlike hardware, software cannot be physically touched. It includes various applications tailored for specific functions, such as web browsers for internet access or games for entertainment.

Hardware and software work together to perform tasks. Hardware provides the physical components such as the processor, memory, and input/output devices; while software delivers the coded instructions that guide those components. Together, they form a system that enables users to perform tasks, run applications, and interact with digital content.

- For example, the user enters input into the computing system using a mouse or keyboard (hardware). The input or data is then processed in the microprocessor or central processing unit (hardware). Once the computing system processes the data following the program rules (software), it generates output.

[2.CSY.1b] Input refers to data, commands, or information that a user provides to a computer system to initiate a task or generate a response. This input can be entered through various input devices such as a keyboard, mouse, touchscreen, microphone, or camera. These devices allow users to interact with the system by typing, clicking, speaking, or capturing images.Output, on the other hand, is the information the computer system produces after processing the input using its hardware and software components. Output devices such as a monitor, printer, or speaker allow users to see, hear, or receive the results of the computer's operations. Together, input and output form a key part of the input–process–output (IPO) model, which describes how computing systems function to receive, process, and return information.

Input Device	Output Device
Keyboard 	**Monitor** 
Mouse 	***Speakers*** 
Touchscreen 	***Printer***  **Image Sources Pixabay**

[2.CSY.1c] A computing system receives input and provides output. Input is the information or instructions entered into a computer using a mouse, keyboard, microphone, cameras, or touch screens. Examples of input include numbers, passwords, lunch numbers, answers, and user choices. Output is information provided by a computer after it does what was asked or instructed by the input that was received. Examples include words appearing on a monitor or screen, printed documents, or spoken words.

[1.CSY.1d] Computers use binary code to communicate and process information. Binary is a special language that computers use to store and process all information, represented by two numbers: 0 and 1. Computers store and process all information using a sequence of binary numbers. These binary digits, or bits form the foundation of all computer memory. Each key on the keyboard has a binary sequence assigned to it. The user pushes keys for letters and numbers. The computer's central processing unit (CPU) reads the binary code assigned to each key and processes the instructions following the program. The central processing unit (CPU) translates the binary code back into the letters and numbers and displays on the screen or via the printer in a readable format for the user.

Examples of computing systems at school:

Cafeteria Meal System:

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

School Media Center:

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

Concepts and Connections

CONCEPTS

Computers process information using binary code to interpret and communicate data between hardware and software. Hardware and software work together to accomplish tasks by allowing the computer to receive input, process data, and produce output.

CONNECTIONS

Within the grade level/course: At this grade level, students describe the characteristics of computing systems including hardware, software, input, and output (2.CSY.1).

Vertical Progression: In Grade 1, students describe how computing components work together to create a computing system (1.CSY.1). By Grade 3, students will model how computing devices within a computing system work (3.CSY.1).

ACROSS CONTENT AREAS

English

- **2.RV.1** Vocabulary development and word analysis: b) use vocabulary across content areas and h) use newly learned words and phrases in discussions and speaking activities.

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 role-play running a grocery store with a "robot cashier" who follows instructions (software) on cards to calculate the total cost (output) of a customer's order (input) using toy food items with price tags. They will practice counting by 2s, 5s, and 10s, and addition/subtraction within 20. Afterward, they will discuss the parts of their "computing system," identifying the hardware (toy food items and cards), software (instructions on cards), input (customer's order), and output (total cost). This activity helps students understand the characteristics of computing systems, including hardware, software, input, and output.

Science

- Students will explore the relationship between input and output in computing systems by comparing it to the effects of heating and cooling on phase of matter. They will examine how energy input leads to transformations in both digital and physical systems, reinforcing the connection between computational processing and scientific phenomena.

- Students will monitor the weather station. The students use a rain gauge (hardware) to collect rain or other precipitation measurements (input). Students will process the amount of rain and precipitation collected in the rain gauge each day. Students will record the rain and precipitation data in inches (software) and then display it on a class chart (output).

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

2.CSY.2 The student will demonstrate an understanding of how to troubleshoot simple hardware and software problems that may occur during use.

a. Propose solutions to simple hardware and software issues.
b. Use appropriate steps to perform simple troubleshooting tasks.

Understanding the Standard

Computing systems might not work as expected because of hardware or software problems. Identifying and describing a problem is the first step toward finding a solution. As early as Kindergarten, students are expected to use developmentally appropriate language to describe a problem when using computing technologies. Examples include, "The computer won't turn on," "The pointer on the screen won't move," or "I lost the web page."

[2.CSY.2] Although computing systems may vary, common troubleshooting strategies can be used with most computing devices. Troubleshooting involves identifying and correcting faults in a computing system. Since computing systems are composed of an interconnected components of hardware and software, troubleshooting strategies may need to address both.

[2.CSY.2a] This standard requires students to propose solutions to simple hardware and software issues.

- Possible solutions to simple hardware issues include checking if the computer is turned on, making sure the monitor is turned on and the screen is lit, checking to see if any power cords are unplugged or cables are not connected, and listening for any unusual sounds.
- Possible solutions to simple software solutions include turning the computer on, looking to see if the computing device is connected to the internet, checking to see if airplane mode is on, and looking for update or warning messages.

[2.CYS.2b] The student is expected to use appropriate steps to perform simple troubleshooting tasks.

Appropriate steps to troubleshooting hardware issues include:

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

Appropriate steps to troubleshoot software issues include:

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

Make a class anchor chart showing troubleshooting steps students are able to complete independently. This might vary from class to class. Add extra steps as students are ready for them.

Troubleshooting My Computer
1. Is my computing device on? If no, turn on the computing device.
2. Is my battery low? If yes, plug the computing device into a power source.
3. Did I sign in? If no, sign in to the computing device.
4. Am I online? If no, sign in to the school's network.
5. Do I have an updated message? If yes, reboot or restart the computing device.

Concepts and Connections

CONCEPTS

Computing devices may not always work as intended and errors can occur during use. It is important for students to understand that these issues are common and can often be resolved through troubleshooting. By using basic troubleshooting strategies, students can identify the cause of the problem and take appropriate steps to resolve it.

CONNECTIONS

Within the grade level/course: At this grade level, students apply troubleshooting strategies to resolve simple hardware and software issues during use, developing problem-solving skills that enhance their ability to use digital and computing tools effectively. (2.CSY.2)

Vertical Progression: In Grade 1, students use accurate terminology to describe when a computing system might not work as expected (1.CSY.2). In Grade 3, students use accurate terminology when troubleshooting problems within a computing system if it is not working as expected (3.CSY.2).

ACROSS CONTENT AREAS

English

- **2.C.3** Integrating multimodal literacies: a) create a simple presentation using multimodal tools that enhance the topic or presentation. (Note: **students should use text to gain information for multimodal.*)

DIGITAL LEARNING INTEGRATION

- **K-2. 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.
- **K-2.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.

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 create picture stories enhanced with sounds using a simple app. They draw or find pictures and add fun sound effects. While building their digital stories, they might encounter small tech problems, like a picture not showing up or a sound not playing. These are opportunities to practice problem-solving, like checking file types or volume levels. Finally, they share their multimedia stories with the class.

Science

- Students may work in groups as "Computer Detectives" solving the "Case of the Curious Computer"! Each group receives a device that won't turn on. First, observe the problem and write down questions about possible reasons. Then, guess (make a hypothesis) what the problem is. Test your guess! Try different things but not taking devices apart. If your guess is wrong, try again. Once you know the problem, think of a solution. Could a new tool or design help?

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 Computing Systems (CSY)

Cybersecurity (CYB)

2.CYB.1 The student will model safe and responsible behaviors when using information and computing technologies.

a. Explain the need for safe and responsible uses of computing technologies.
b. Create a flowchart to illustrate the process for reporting inappropriate use of technology at school or at home.
c. Demonstrate and model safe and responsible behaviors when using computing technologies and online communication.

Understanding the Standard

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 and divisions have rules on the appropriate use of technology.

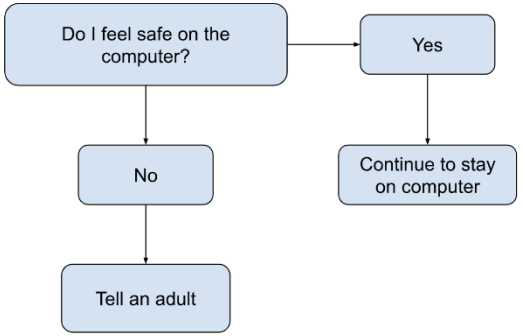
[2.CYB.1a] Computing technologies refer to a broad range of devices and tools designed to process, store, and transmit information. These tools assist users process information and perform tasks. These technologies enable students to access information, communicate, create content, and solve problems.

Safe and responsible use of computing technologies is essential for protecting personal information and ensuring privacy online. Following rules and acceptable use polices help protect personal data, individuals, online content, and network systems. Appropriate use of technology should be reviewed frequently with students. Students must understand that safe and responsible use of computing technologies helps keep them and their information safe.

Cybersecurity practices, such as password protection and visiting trustworthy sites are foundational safeguards in keeping one safe. Additional practices include logging off when finished using a device or application and not clicking on unknown links or pop-ups, which can lead to unsafe websites. Students should ask an adult before downloading or sharing information online.

[2.CYB.1b] Flowcharts are diagrams that show a sequence of steps to complete a task or solve a problem. Flowcharts use arrows and symbols to show the process or flow of actions in order. Flowcharts can be used to show how to report inappropriate technology use.

Consider the following examples:



A flowchart is like a map that shows you what to do. It helps you know what steps to take if you don't feel safe using technology, and you need to know what steps to take next. If you notice inappropriate use of technology at home or school, such as a pop-up message that you did not click on, you need to tell an adult. You need to tell them what happened and when it happened.

[2.CYB.1c] By practicing responsible digital habits, students can create a safer online environment and become responsible digital citizens. Safe and responsible use expands beyond the responsibility of computing devices and includes the protection of an individual and an individual's online reputation. A digital footprint is one's online reputation that is comprised of public and private information.

Similar to the norms in the physical environment, kindness, respect, and accountability are essential while online and communicating with others. It is not appropriate to cause harm, mean, or exhibit bullying behaviors. Cyberbullying can exist online, and it can have a lasting effect on those involved, making it essential for students to recognize the power of their words and actions. Students should also be aware of online risks that are similar to the physical environment and avoid sharing personal information with strangers and seek guidance from trusted adults for all actions and activities online. When students witness or experience inappropriate behaviors online students should know how to respond appropriately, what trusted adulted to speak to, and how to report such behaviors.

Concepts and Connections

Concepts

Safe and responsible use of computing technologies is a key aspect of being a responsible digital citizen. Digital citizenship involves making good choices online, showing respect to others, and utilizing technology in ways that are mutually beneficial.

Connections

Within the grade level/course: At this grade level, students explain the need for safe and responsible uses of computing technologies, create a flowchart to illustrate the process for reporting inappropriate use of technology at school or at home, and demonstrate and model safe and responsible behaviors when using computing technologies and online communication (2.CYB.1).

Vertical Progression: In Grade 1, students describe safe and responsible uses of computing technologies based on the school rules and acceptable use policy (AUP), demonstrate safe and responsible behaviors when using computing technologies and online communication, discuss the process for reporting inappropriate technology use at school or home, classify appropriate and inappropriate uses of technology at school or at home and explain the consequences of inappropriate uses of computing technologies (1.CYB.1). By Grade 3, students Identify and distinguish personal information that should be private, describe the importance of using a strong password, and create and use strong passwords to protect private information (3.CYB.1).

Across Content Areas

History and Social Science

- **2.1** The student will apply history and social science skills to distinguish between the rights and responsibilities that individuals have in the United States including but not limited to e) respecting and following laws and f) practicing honesty and trustworthiness.

Digital Learning Integration

- **K-2.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.
- **K-2.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.
  D. Publish or present content that customizes the message and medium for the intended audiences.
- **K-2.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.

English

- Students will combine informative writing with digital citizenship by researching and writing about a chosen online safety topic, such as cyberbullying. They use facts and examples in their informative texts and share their writing with the class, discussing how this knowledge helps them be responsible and safe online.

History and Social Science

- Students will learn about being "Digital Superheroes" by discussing online scenarios, like finding a cool game on a website. They explore different choices and their consequences, such as whether it's okay to download it without asking a grown-up. They decide what a responsible "Digital Superhero" would do. The class then creates a "Digital Superhero Pledge" which outlines rules for safe and kind online behavior, covering topics like protecting personal information and asking for help from a trusted adult.

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

2.CYB.2 The student will explain the importance of using passwords to protect private information.

a. Identify and classify passwords as strong or weak.
b. Explain how a strong password helps protect the privacy of information.
c. Explain the risk of sharing passwords.

Understanding the Standard

Connecting devices to a network or the Internet offers many benefits, including access to information, communication tools, and online learning resources. However, this connectivity also increases the risk of exposing personally identifiable information (PII) such as a student's name, phone number, or home address. To safeguard this information, passwords are commonly used as a first layer of protection against unauthorized access to devices, accounts, and data.

Because automated tools can be used by malicious actors to guess passwords, it is essential to use strong passwords that are complex, unique, and difficult to predict. Strong passwords typically include a combination of uppercase and lowercase letters, numbers, and special characters, and avoid personal details or common words.

[2.CYB.2ab] Passwords help protect the privacy and security of personal information by preventing unauthorized access to devices and accounts. A password can be categorized as strong or weak depending on how easy it is to guess. A strong password is long, unique, and includes a mix of letters, numbers, and symbols. It should not be a word that one can find in a dictionary nor personal information such as your birthday or your pet's name. Weak passwords are passwords that are easy for others to guess or decode. Examples of weak passwords includes but not limited to "password" or "yourname123". Weak passwords do not provide data protection and can be guessed by people or computer programs that are designed to decode "crack" passwords. *At the elementary level, students often use passwords that are created for them by teachers or school systems. These passwords may be simpler and may not always follow best practices for strong password protection, but they serve as an introduction to the concept of securing personal information.*

A strong password should include:

1. Uppercase and lowercase letters.
2. Numbers.
3. Symbols.
4. At least 8 characters. (Note: T*his is age appropriate and should increase in length and complexity as students mature.*)

A password should not include:

1. Words from a dictionary.
2. The same password twice.
3. Personal information.

Consider the following examples:

A strong password might be ‘PxHn32s!@5’
A weak password might be ‘12345678’

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

[2.CYB.2c] Maintaining password security is essential to protecting personal information and online security. Sharing passwords can significantly increase the risk of unauthorized access to computing devices and accounts. This could lead to the potential misuse of accounts or personal information.

Concepts and Connections

CONCEPTS

A strong and secure password helps protect information that is private, like your name or where you live. Strong passwords use a mix of letters, numbers, and symbols. Keeping passwords private helps keep personal, private information safe and secure.

CONNECTIONS

Within the grade level/course: At this grade level, students identify and classify passwords as strong or weak, explain how a strong password helps protect the privacy of information, and explain the risk of sharing passwords (2.CYB.2).
Vertical Progression: In Grade 1, students described the purpose of usernames and passwords, discussed how passwords are private information and are used to protect the privacy of information (1.CYB.2). In Grade 3, students describe how authentication and authorization protect private information, identify multiple authentication methods, and discuss the security risk posed by not having a strong password (3.CYB.2).

Across Content Areas

History and Social Science

- **2.1** The student will apply history and social science skills to distinguish between the rights and responsibilities that individuals have in the United Staes including but not limited to e) respecting and following laws; f) practicing honesty and trustworthiness; and respecting the rights, beliefs, and opinions of others.

Digital Learning Integration

- **K-2.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 explore and discuss how people throughout history have protected personal information, like the use of diaries and secret codes.

Mathematics

- Students become "Secret Agents" learning about passwords by exploring patterns. They create codes using repeating patterns, then connect the idea of complex patterns to strong passwords, understanding that complex passwords, like complex patterns, are harder to guess and protect personal 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.

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)

2.DA.1 The student will analyze data to make decisions with or without a computing device.

a. Collect and record numeric and non-numeric data and describe possible patterns.
b. Create questions that can and cannot be answered by the data.
c. Analyze data to draw conclusions and make decisions.

Understanding the Standard

People use their senses to observe and gather data about the world around them. Data consists of individual pieces of information collected about people, objects, or phenomena. These data can be systematically recorded in tables and then translated into visual representations, such as object graphs or picture graphs, to support interpretation and comparison. Everyday digital devices including cell phones, smart appliances, and modern vehicles are equipped with sensors and embedded computing systems that automatically collect and display data from their surroundings. This automated collection over time enables users to detect patterns, identify trends, and make informed decisions based on the data.

After data is collected and organized into a visual format such as a chart or graph, it can be analyzed to detect patterns, trends, or anomalies. This process may involve comparing numerical values, tracking changes over time, or exploring relationships among variables. Through data analysis, students learn to draw evidence-based conclusions, generate predictions, and respond to questions grounded in observed data. These analytical skills are fundamental to computational thinking and also enhance critical reasoning and problem-solving across academic disciplines.

[2.DA.1a] The collection and use of data about individuals and the world around them is a routine part of daily life and significantly influences personal decisions, societal trends, and technological development. Data can be classified as either qualitative (non-numeric) or quantitative (numeric) depending on its form and how it is used. For example, identifying the color of an item represents qualitative data, while determining how many students in a class prefer vanilla ice cream is an example of quantitative data. Both types of data can be collected through surveys, observations, or sensors and are often organized for analysis in charts, graphs, or databases. Understanding the distinction between data types helps students interpret information accurately and apply it meaningfully in various real-world and academic contexts.

Everyday computing devices can be used to collect and display data over time.

- Examples include cell phones, digital toys, and cars.
- These can collect and display data from their surroundings.

[2.DA.1b] Formulating questions is a foundational step in the data analysis process, guiding what data is collected and how it will be used. Effective data questions are clear, measurable, and aligned to a specific purpose or inquiry. For example, asking "How many students prefer walking to school over taking the bus?" directs the collection of categorical data related to student transportation preferences. To create strong data questions, students must consider the variables involved, the type of data needed (numeric or non-numeric), and how the results will be analyzed or represented. This process builds critical thinking and helps students engage in purposeful data collection that leads to meaningful conclusions and evidence-based decision making.

Questions that can be answered when analyzing data.

- What is the total or sum of the data?
- What is the average or typical value (mean, median, or mode)?
- Which item or category appears most often ?
- Are there any patterns or trends in the data?
- What predictions can we make based on these patterns?
- Are there any outliers or anomalies that stand out?
- How do different data points compare or relate to one another?

Questions that cannot be answered when analyzing data.

- Why was that the most popular choice?
- What is your opinion?
- What do you believe or feel about this topic?

[2.DA.1c] Computers are powerful tools for working with information and data. They can be used to sort, classify, and organize data into categories similar to grouping items, such as toys, based on color or type. Computers can also analyze data to detect patterns, such as identifying whether there are more red items than blue ones. In addition to processing information, computers can generate visual representations, like charts or graphs, to help users better interpret and understand the data.
To do this, computers rely on software programs, special sets of instructions that tell the computer how to process and display data. Some software is designed to store large volumes of information, while other programs specialize in data visualization, transforming numbers and categories into colorful, easy-to-read graphs and models that support decision-making and learning.

Once data has been collected and organized into a chart or graph, it can be analyzed to draw conclusions, answer specific questions or make predictions. You can use data to make choices even without a computer. We can use graphs to show patterns in data and to help us answer questions. Graphs can help us see things that we might not see in a list of numbers. For example, we can use a graph to see how many people

like different colors. We can use a graph to see if there is a pattern between two things, like how many hours of sunlight a plant gets and how tall it grows. (Mathematics 2.PS.1)

Students should practice analyzing the data, asking and answering questions, and drawing conclusions.

Concepts and Connections

CONCEPTS

Analyzing data allows for informed decision-making, whether using computing devices or not. By collecting and recording numeric and non-numeric data, such as measurements or observations, patterns can be identified. Students can ask questions about data to help understand and make decisions. This process strengthens critical thinking and illustrates how data can be used to solve problems.

CONNECTIONS

Within this grade level: At this grade level, students collect and record numeric and non-numeric data and describe possible patterns, create questions that can and cannot be answered by the data, analyze data to draw conclusions and make decisions (2.DA.1).
Vertical progression: In Grade 1, students identify data formats used for various purposes, including audio, images, text, and video, and explore and identify computing devices that collect, store, and/or display data (1.DA.1). By Grade 3, students formulate questions that require the collection or acquisition of data, gather, organize, sort, and store data, examine a labeled dataset to identify potential problems within the data, discuss how data discrepancies or problems impact predictions and results, and draw conclusions and make predictions based on observed data (3.DA.1).

ACROSS CONTENT AREAS

Mathematics

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

Science

- **2.1** The student will demonstrate an understanding of scientific and engineering practices by a) asking questions and defining problems and c) interpreting, analyzing, and evaluating data. *2.1 standard is integrated within science content and not taught in isolation. Potential science concepts to apply 2.1 include: 2.2 (force, motion, energy), 2.3 (matter), 2.4 (plant life cycle and animal life*

cycle) and 2.5 (living things).

- **2.4** The student will investigate and understand that plants and animals undergo a series of orderly changes as they grow and develop. Key ideas include b) plants have life cycles.

DIGITAL LEARNING INTEGRATION:

- **K-2.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  A. Formulate problem definitions suited for technology assisted methods such as data analysis, modeling and algorithmic thinking in exploring and finding solutions.
  B. Collect data or identify relevant data sets, use appropriate technologies to analyze them, and represent data in various ways to facilitate problem-solving and decision-making.

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 look at pictographs and bar graphs to understand and talk about the data.
- Students ask and answer questions. For example, how many data points are in total? How many are in each category? How many are in one category compared to another?
- Students use pictograph symbols that represent 1, 2, 5, or 10 pieces of data.
- Students use bar graphs scales with increments of 1, 2, 5, or 10.

Science

- Students grow bean plants, measuring them regularly and graphing their growth. They analyze the data to understand the plant's life cycle and then design improved growing systems based on what they learned, like adding a light or a self-watering feature.

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

2.DA.2 The student will manipulate data, create representations, and evaluate data to solve a problem.

a. Create charts, graphs, and models using abstraction to represent data.
b. Analyze data visualizations to draw conclusions.
c. Propose and evaluate a solution to a problem or question based on data and/or data visualization.

Understanding the Standard

Scientists, computer scientists, mathematicians, and programmers construct and use models to better conceptualize and understand events under investigation or to develop a possible solution to a proposed problem. Models include diagrams, physical replicas, mathematical representations, analogies, and computer simulations. Models are used to represent a system (or parts of a system) under study, to aid in the development of questions and explanations, to generate data that can be used to make predictions, and to communicate ideas to others. Students are expected to create simple models. These models may be created using a computing device.

[2.DA.2a] Charts and graphs are like pictures that show us information. They help us see numbers and information in a way that is easy to understand. For example, a bar graph uses bars of different sizes to show us how much of something there is. A line graph uses lines to show how something changes over time. (Mathematics 2.PS.1 abcde)

A model emphasizes the most important aspects of a system, helping students focus on key data. Through abstraction, unnecessary details are removed to simplify the system and highlight the essential elements. This process makes it easier to understand the system and focus on what is most relevant. When creating a model, students should prioritize key information that will help solve a problem or answer a question.

Models can be used to represent systems or processes that are not directly observable, such as how a computer operates or how data is transmitted across the internet. Through modeling, students can simulate ideas, analyze relationships, make predictions, and develop a deeper understanding of abstract or complex computing systems. Models also provide a low-risk environment for experimentation, allowing students to visualize inputs, processes, and outputs without needing to access actual hardware or infrastructure. This supports both conceptual understanding and the development of transferable problem-solving skills.

- **Types of models**: Models can take many forms, including diagrams, physical replicas, mathematical representations, analogies, and computer simulations. Each model is used to represent or explain a system, process, or concept in a simplified and accessible way.

- **Example model**: A map of a classroom that includes key elements such as desks, chairs, and the teacher's desk. This model helps visualize the spatial arrangement without needing to enter the physical room.

- **Use of abstraction**: The classroom map shows only essential components (e.g., furniture placement and room layout), leaving out unnecessary details like individual pencils, books, or decorations. This abstraction helps students focus on the overall structure and function rather than irrelevant specifics.

Modeling is crucial as it enables the representation of real-world situations, and allows for making predictions, testing hypotheses, and solving complex problems using computer science tools and concepts. Modeling may be a way to represent numbers, natural events, the visualization of data, or behavior.

Manipulating data provides students with the ability to organize, sort, adjust, and compare data in ways that help them draw conclusions. This could include sorting numbers, counting items, or comparing different sets of data to identify patterns and better understand data visualizations.

[2.DA.2b] Analyzing data requires students to look closely at the data to identify patterns, compare different groups, and find outliers. During this process students should ask questions to deepen their understanding and better explain what the data is showing. After analyzing, student can draw conclusions and make decisions by using the information from the charts, graphs, and models.

[2.DA.2b,c] Data visualization is a way to show information using pictures like charts, graphs, or symbols to make the data easier to understand. Presenting data in this format helps people see patterns and compare information quickly. For example, we can use a line chart to show how much money a company makes each year and use this information to make informed decisions. Evaluating data and drawing a conclusion is easier than looking at a table of numbers and then drawing a conclusion.

Data analysis involves examining data to uncover meaningful patterns, trends, and insights that can help inform decision-making. By organizing and interpreting this data, we can identify important information that might not be immediately obvious. This process allows us to draw conclusions, make predictions, and develop strategies based on the evidence provided by the data. Common examples of data analysis include but are not limited to student performance, survey results, website traffic, weather data, and sales data.

Concepts and Connections

CONCEPTS

Students organize and manipulate data using tools like charts, graphs, and models to deepen their understanding. Models, which can be physical or digital, represent real-world objects or processes and simplify complex systems. These models provide a clearer insight into how systems function and how various components interact. By using models, students can visualize abstract concepts, analyze data relationships, and make predictions based on their findings.

CONNECTIONS

Within this grade level: At this grade level, students create charts, graphs, and models using abstraction to represent data, analyze data visualizations to draw conclusions, and propose and evaluate a solution to a problem or question based on data. (2.DA.2).
Vertical progression: In Grade 1, students collect and organize data with or without a computing device, create tables, picture graphs, and models using abstraction, identify patterns and describe trends in data visualizations of various formats. Students also use data to answer questions, draw conclusions, and make predictions (1.DA.2). By Grade 3, students create charts and graphs based on data collection, and analyze data to identify patterns, draw conclusions, and make predictions (3.DA.2).

ACROSS CONTENT AREAS

English

- **2.W.1** Modes and Purposes of Writing: b) write informative/explanatory texts that introduce a topic and develop the ideas with facts and examples.

Mathematics

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

DIGITAL LEARNING INTEGRATION:

- **K-2.CT** Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods, including those that leverage assistive technologies, to develop and test solutions.
  A. Formulate problem definitions suited for technology assisted methods such as data analysis, modeling and algorithmic thinking in exploring and finding solutions.
  B. Collect data or identify relevant data sets, use appropriate technologies to analyze them, and represent data in various ways to facilitate problem- solving and decision-making.
  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.
- **K-2 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.
  C. Use appropriate technologies to develop, test, and refine prototypes as part of a cyclical design process.

Opportunities for Computer Science Integration

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

Mathematics

- Students analyze data from a class survey by organizing the information into a model.
- Students gather data about how many hours of sleep they get each night for a week and represent using a bar graph. Based on data, students may need to propose solutions to get the recommended hours of sleep.

Science

- Students gather data about local weather patterns over a week and create a visual model to represent the temperature and weather conditions each day.
- Students interpret data from plants' growth and discuss how different factors like sunlight or water may have influenced the results and propose what could help the plant grow.

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)

2.IC.1 The student will examine the positive and negative impacts of how using computing technologies has changed the way people live, work, and interact.

a. Identify current uses of computing/emerging technologies and discuss how they impact society.
b. Compare and contrast appropriate and inappropriate online behaviors that apply in the physical environment and the online environment.
c. Model healthy habits for using computing technologies.

Understanding the Standard

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

[2.IC.1a] Computers have greatly changed the way we live, work, and interact, bringing both positive and negative impacts. Some positive impacts are they help us perform tasks more efficiently, such as finding information quickly, shopping online, playing educational games, and watching videos. They make certain tasks easier by automating repetitive processes, allowing us to quickly access information, connect with others instantly, and streamline work in areas like education, healthcare, entertainment, and banking.

Computers enhance productivity by facilitating communication through emails, creating presentations, and managing important data and schedules. They also allow for remote work, offering flexibility and a better work-life balance. By integrating AI tools, such as voice assistant, individuals can enhance their productivity, improve communication, and create a more organized and supportive work environment. The use of AI tools still requires human oversight and an understanding of how AI works to ensure it is used effectively and responsibly.

Computers have greatly improved productivity and revolutionize how we communicate. They allow us to send messages, make video calls, and share photos with friends and family across any distance, helping to maintain connections. This connectivity has given rise to concerns like cyberbullying and privacy risks. Cyberbullying is a form of bullying that occurs when online communications are sent that are intimidating or threatening in nature. To fully harness the benefits of technology while mitigating its drawbacks, it is essential to approach its use thoughtfully and responsibly.

The increased reliance on technology introduces challenges, such as reduced face-to-face interactions and physical activity. This shift can affect social skills, as people may have difficulties with personal communication and building strong relations.

[2.IC.1b] Online behaviors, such as showing respect for others, communicating concisely and clearly, and following rules are equally important in the online environment as they are in the physical environment. People should practice kindness, be mindful of privacy, and avoid harmful behaviors, whether interacting fact-to-face or virtually. Maintaining good online behaviors helps create safe, positive, and respectful digital spaces, as it would in a physical environment.

Appropriate Online Behavior	Inappropriate Online Behavior
• Using a computer to find fun facts for a school project. • Playing educational games that teach math and reading. • Video calling grandparents and other relatives who live far away.	• Spending too much time on screens and not enough time playing outside. • Seeing or sharing things online that aren't safe or suitable. • Missing out on talking to friends and family in person. • Arguing online instead of discussing differences with kindness and respect.

[2.IC.1c] Responsible behavior should always be used when working with computers, such as not sharing login information, keeping passwords private, and logging off when finished. These behaviors remain essential whether students are using technology at school or in any other setting.

Modeling healthy habits for using computing technologies involves demonstrating balanced and responsible use of devices. This includes setting boundaries for screen time, taking regular breaks to avoid eye strain, and encouraging physical activity. It's also important to show how to use technology for productive purposes, like research and communication, while being mindful of online safety and privacy. By building these habits early, students develop the skills they need to use computing technologies responsibly and effectively in all areas of their lives.

Concepts and Connections

CONCEPTS

Computing technologies have transformed the way people live, work, and interact. This has increased productivity, global communication, and increased the amount of information people have access to. However, these advancements bring challenges, including the potential for inappropriate behaviors, privacy concerns, and the impact of excessive screentime on physical health and overall well-being.

CONNECTIONS

Within this grade level: At this grade level, students identify current uses of computing/emerging technologies and discuss how they impact society, compare and contrast appropriate and inappropriate online behaviors that apply in the physical environment and the online environment, and model healthy habits for using computing technologies (2.IC.1).
Vertical progression: In Grade 1, students determined when tasks should be completed with or without computing devices, described how computing devices are used in communication, and described healthy habits for using computing technologies. (1.IC.1). In Grade 3, students identify computing technologies that have changed the world, examine and explain how computing technologies influence and are influenced by culture, and identify social and ethical issues related to the use of computing technologies (3.IC.1).

Across Content Areas

English

- **2.RI.1** Key Ideas and Confirming Details: a) ask and answer literal and inferential questions (who, what, where, when, how, and why) about key details in a text; and c) differentiate facts from opinions within a text. (Note: *Reading text should examine the positive and negative impacts of computing.*)

History and Social Science

- **K.1** The student will apply history and social science skills to practice citizenship in the classroom by a) taking responsibility for one's actions, b) practicing honesty and showing kindness to oneself and others, c) recognizing the purpose of rules and practicing self-control, d) caring for one's personal property and respecting other students' property, and e) taking turns, sharing, and working well with others for the good of everyone.

Digital Learning Integration

- **K-2.DC** Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act in ways that are safe, legal, and ethical.
  A. Cultivate and manage their digital identity and reputation and are aware of the permanence of their actions in the digital world.
  B. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.
- **K-2 KC** Students critically curate a variety of digital resources using appropriate technologies, including assistive technologies, to construct knowledge, produce creative digital works, and make meaningful learning experiences for themselves and others.
  A. Plan and employ effective research strategies to locate information and other digital sources for their intellectual or creative pursuits.

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 write a short essay or letter, describing what their day would be like if there were no computing technologies, like no computers or no phones. Students could explain how their life, school, and talking to friends might change.

History and Social Sciences

- Students create a timeline showing the major communication inventions and how they changed the way people communicated over time.
- Students will compare the process used to find books in the library and using the Internet.
- Students create visual drawings on responsible and respectful behaviors in both the physical classroom and online environment.

Science

- Students discuss how technology like smartphones and computers uses resources like energy, metals, and minerals.
- Students discuss how they can help the environment by recycling old electronics or turning off devices when not in use.

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

2.IC.2 The student will explain the need to balance screen time and other activities.

a. Discuss appropriate times and places for screen use.
b. List and describe alternatives to screen time.

Understanding the Standard

[2.IC.2a] Balancing screen time is essential for students' healthy development. Too much screen time can affect sleep and make it harder to focus. It is vital to identify appropriate use of computational technologies and limit use when unnecessary. Limiting screen use encourages more physical activity and face-to-face interactions which are important for health and social skills.

[2.IC.2b] Being aware of alternatives to using computational devices help students make informed decisions on how to spend their time. Engaging in offline activities prompt physical health, social skills, and emotional development. Engaging in offline activities promotes physical health, social interaction, and emotional regulation. Encouraging students to reflect on their media habits fosters self-awareness and responsible decision-making. Over time, these practices support a balanced relationship with technology and contribute to long-term digital well-being.

Topic	Examples
Appropriate Times and Places for Screen Use	• During school or homework time • In common areas like the living room • On the weekend • Avoid during meals • Avoid before bedtime • Avoid family time or outdoor play
Alternatives to Screen Time	• Reading Books: Enjoying stories and learning new things • Playing Outside: Getting fresh air and exercise through games or sports • Arts and Crafts: Being creative with drawing, painting, or making crafts • Board Games and Puzzles: Having fun and challenging the mind • Spending Time with Family and Friends: Engaging in conversations and activities together without screens • Listening to Music: Enjoying and dancing to favorite songs • Gardening: Planting flowers or vegetables and taking care of them • Building: Design with blocks and create structures using imagination

Concepts and Connections

Concepts

Students benefit from maintaining a healthy balance between screen time and offline activities. Engaging in activities such as outdoor play, reading, and creative expression supports not only physical health and social-emotional development, but also cognitive growth. Developing self-regulation around technology use is a key aspect of fostering digital well-being.

Connections

Within this grade level: At this grade level, students discuss appropriate times and places for screen use, list and describe alternatives to screen time (2.IC.2).
Vertical progression: In Grade 1, students identified daily routines and activities that can be completed with or without screens and classified the different uses of screen time as learning, entertainment, or communication (1.IC.2). By Grade 3, students define and describe screen time, explain the importance of responsible screen time management, and discuss how screen time choices affect one's personal health and interactions with others (3.IC.2).

Across Content Areas

English

- **2.C.1** Communication, Listening, and Collaboration: a) participate in a range of collaborative discussions (one-on-one, in groups, and teacher-led) on grade two topics and text.

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 role-play "digital dilemmas" about balancing screen time and other activities, like choosing between homework and online games. After each role-play, the class discusses communication and problem-solving strategies, then creates guidelines for healthy technology use.

Mathematics

- Students track screen time and other activities by creating graphs to visualize their time use. Students analyze this data to discuss a healthy balance and create guidelines for responsible technology use.

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

2.IC.3 The student will explain how computing technologies have an impact on the workforce.

a. Explain how computing technology is used in various careers.
b. Identify skills needed for careers that use computing technologies.
c. Discuss how computing technologies have changed the workplace.

Understanding the Standard

Computing technologies are used in various careers to improve productivity, solve problems, and enhance innovation. This has ranged from the use of computing technologies in healthcare to analyzing medical data to the education where computing technologies can be used to foster collaboration and enhance learning.

[2.IC.3a] Computers play a vital role in various careers today. They are used in healthcare to manage patient records, in finance to analyze data and predict trends, in education to create interactive learning experiences, and in engineering to design and test new products. In the creative industries, computers are essential for graphic design, video editing, and music production. Additionally, computers facilitate remote work, allowing professionals to collaborate and communicate from anywhere in the world.

[2.IC.3b] To work effectively with computers, individuals need a range of skills. These skills require digital, media, and AI literacy. Digital literacy is the ability to use and understand digital devices and software. Media literacy consists of understanding, analyzing, and creating content in various digital media formats. AI literacy is the ability to understand AI concepts both what it is and how it works, including AI tools capabilities, limitations, and ethical considerations. Here is a table outlining some careers and the computing skills required:

Field of Study	Early Example	Modern Example
Astronomy	Astronomers use telescopes to look at stars and write down their findings.	Astronomers and scientists use computers to monitor stars and planets. Computers make it possible to analyze changes that may occur over time.
Engineering	Engineers draw plans for buildings and other structures by hand or by slide rulers.	Engineers use computer aided drafting software to aid in designing structures.
Healthcare	Nurses recording patient information and vitals on paper charts.	Nurses record patient information and vitals using digital charts.
Transportation	Travelers and motorists use paper maps and atlases to plan trips.	Travelers and motorists utilize GPS and online traffic apps to plan trips and avoid traffic.

Communication	Individuals send letters, cards, and pictures by mail.	Individuals write and send emails. Individuals make online video calls.
Education	Teachers use paper gradebooks and calculators to store and calculate grades.	Teachers use computers for assessments, grading, storing grades, and other student information.
Research	Scientists write down experiment results.	Scientists utilize computers to analyze and store data from experiments.
Commerce	Shopkeepers and businesses write down sales and send out invoices by mail.	Store managers and businesses use computers to communicate sales, capture point of sale transactions, and invoice customers.
Military	Armed forces use paper maps for planning and logistics.	Armed forces use computers for tracking, strategic planning, and logistics.

[2.IC.3c] Over the past 100 years, computers have become much more powerful and essential in many jobs. Computers help people in their communities and at work by connecting with others and improving learning. They have evolved from large machines to small devices like smartphones. They have transformed industries such as healthcare and transportation by providing information and connecting people globally.

People have always used devices to assist in computation. They can help in the collection, storage, or manipulation of data. Early computers used mechanical components to perform calculations. In the early 1800s, the first programmable computers were created. They were limited in their capability and relied heavily on people to do more complex computation. These people were referred to as “computers” due to their similar role. Many of these “computers” were women who were employed in commerce, government, military, and research establishments.

One early computer, the ENIAC, built in the 1940s, was very large and used a lot of electricity. It paved the way for the small, powerful computers we use today. Computers make tasks easier and faster. They allow quick communication, use software to streamline project management, and analyze data efficiently. Without computers, all aspects of daily life would be very different, including school, home, industry, and commerce.

While computing technologies have improved efficiency in many careers, the rapid evolution of technologies requires people to constantly learn new skills and tools. This can lead to people feeling tired or overwhelmed. As technology evolves, some jobs may change or disappear, and new jobs will be created. It is important to learn new skills so people can continue working.

Concepts and Connections

Concepts

Computing technologies are used in a variety of fields and their use has revolutionized industries. It is important that students understand the impact and role computing technologies have had in the workplace.

Connections

Within the grade level/course: At this grade level, students explain how computing technology is used in various careers, identify skills needed for careers that use computing technologies, and discuss how computing technologies have changed the workplace (2.IC.3).

Vertical Progression: In Grade 1, students identified tasks that can be completed with or without computing technologies and discussed the related advantages and disadvantages. Students described how the appropriate use of computing technologies can improve efficiency, and listed computing technologies used in various careers (1.IC.3). By Grade 3, students research computing technology careers, and describe the impact careers in computing technology have on society (3.IC.3).

Across Content Areas

English

- **2.RI.A** Ask and answer literal and inferential questions (who, what, where, when, how, and why) about key details in text
- **2.RI.B**. Retell key details of texts that demonstrate an understanding of the main topics and texts
  (Note: Standard is aligned if students read text about computing technologies and the impact on the workforce)

History and Social Science

- **2.13** The student will apply history and social science skills to understand basic economic principles by a) identifying natural resources (e.g., water, soil, wood, coal), human resources (i.e., people at work), and capital resources (e.g., machines, tools, computers, buildings).

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 explore how technology is changing jobs, brainstorming future careers like "Drone Traffic Controller." They research a chosen job, creating presentations for a "Future Jobs Fair" to showcase the skills, technology, and importance of these potential tech-related careers.
- Students will discuss how computing technologies like computers, cash registers, and robots have changed the way people work.

Science

- Students explore how technology helps us track weather and prepare for storms, then connect this to how technology impacts jobs like weather forecasting and other careers. They draw weather tools and explain how similar technology is used in different professions, understanding how technology shapes the future workforce.

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 Computing Practices:

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

D. Fostering Digital Literacy Practices:

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

Back to Impacts of Computing (IC)

Networks and the Internet (NI)

2.NI.1 The student will demonstrate the use of the Internet in gathering information to accomplish a task.

a. Explore ways information is organized and shared on the Internet.
b. Gather information from the Internet.
c. Summarize collected information using own words.

Understanding the Standard

The Internet is the underlying infrastructure that connects devices. The internet gives people access to the web. The web is a system of interconnected resources, where information can be found on various platforms. The world wide web acts as a digital library, providing access to a wide range of data and information.

- The Internet is like a vast network of roads and highways. These roads connect different places, allowing vehicles (data) to travel from one location to another. The Internet is the infrastructure that makes communication and data transfer possible. Consider the World Wide Web (www) as the houses and buildings along these roads. Each house represents a website, and the buildings are the web pages within those websites. The World Wide Web is a collection of information that is accessed via the Internet.

People use the Internet to gather information and organize information in many ways, including the use of search engines, watching online videos, reading articles and websites, and communicating with others to accomplish a task.

[2.NI.1a] Information on the Internet is organized and shared on various platforms, making it accessible and easy to navigate. Websites, databases, and cloud storage systems categorize information using tags, keywords, and links. This allows users to quickly find the data or information they need. Online content is often shared using hyperlinks, file-sharing services, and social media.

[2.NI.1b] Students should gather information from trusted, appropriate websites. They should ‘ask’ clear, specific questions and visit multiple reliable sources to ensure accuracy of information. It's important to evaluate the credibility of sources by checking the author's credentials, publication date, and the type of website.

To effectively find information, students can use search engines by typing questions into search bar, using the microphone to ask questions, or enter relevant keywords. For example, elementary students might use keywords like "animal habitats", "famous explorers", or "how plants grow” or type/ask questions like “where do frogs live?” or “what do penguins need in their habitats?”.

[2.NI.1c] When researching online information, students should be able to paraphrase and summarize ideas and support those with evidence, examples, and details. They should practice notetaking to capture key ideas and rephrase them in their own words to enhance understanding. Additionally, students begin to understand the importance of expressing ideas in their own words, and the need to avoid plagiarism when writing or presenting ideas. This sets the foundation for later grades. This approach helps develop essential research skills and critical thinking, preparing students for future academic success. (English 2.R.1 abcde)

Information	Examples
Research	Encourage students to use search engines to find information online. Ensure they use keywords that are relevant to their topic. For example, they might search for "rainforest animals," "famous inventors," or "simple science experiments".
Share Information	Students can share what they know by talking to others, writing about their experiences on the computer, and drawing pictures to illustrate their learning. They can use apps to create presentations to share their new knowledge.
Organize Information	Students can use folders and labels to keep their work organized. They can find information on websites and online libraries, where it is already sorted by topic. Additionally, graphic organizers, such as webs or charts, can assist in visualizing information.
Check Information	Students should make sure information comes from a trustworthy source such as .gov or .edu. Students should check the date to make sure the information is up to date.
Summarize Information	Students should take notes on important ideas in their own words and use pictures to remember the information. Students should give credit to the original author of the text and photos.

Concepts and Connections

CONCEPTS

The Internet provides access to the world wide web, where people communicate and gather or share information. To find reliable and accurate information, it is important to use best practices such as specific keywords and check the credibility of online sources. After gathering information, summarizing key points in one's own words helps one to better understand and apply what was learned.

CONNECTIONS

Within the grade level/course: At this grade level, students explore ways information is organized and shared on the Internet, gather information from the Internet, and summarize collected information using their own words (2.NI.1).
Vertical Progression: In Grade 1, students described how the Internet can be used to gather information and explained ways people communicate using computing devices and the Internet (1.NI.1). By Grade 3, students will differentiate between a network and the Internet, identify the components of a computing network, describe how a computing device connects to a network, and identify ways networks are used to transmit information (3.NI.1).

ACROSS CONTENT AREAS

English

- **2.R.1A** Identify a topic and generate questions to explore the topic
- **2.R.1B** Locate information in reference texts, electronic resources, interviews, or provided sources.

DIGITAL LEARNING INTEGRATION:

- **K-2.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.

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 summarize key points and use keywords to search for reliable online sources.
- Students use the Internet to gain access to online resources to research and organize information into a clear, written format.

History and Social Science

- Students will explore a significant historical event, such as the American Revolution, by gathering primary sources from trustworthy websites.

- Students research major contributors to the field of computer science, such as Ada Lovelace or Alan Turing and summarize their findings in a written or visual format that highlights key developments.

Science

- Students research animal habitats using targeted keywords to find reliable online sources.
- Students explore the water cycle by gathering information and summarize the stages of the water cycle and create a visual diagram or written explanation.

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

Back to Networks and the Internet (NI)

Appendix A

K-5 Computer Science Skills and Practices Continuum

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

A. **Fostering Collaboration in Computing Practices**
1. **Building Relationships and Norms:** • **K-2**: Students work collaboratively with others. Students take turns in different roles on the project. • **3-5**: Students work collaboratively with others. Students practice assigning roles within their teams and recognize group member strengths.
2. **Include Multiple Perspectives:** • **K-2:** Students differentiate their technology preferences from the technology preferences of others. Students will be presented with perspectives from people with different backgrounds, ability levels, and points of view. • **3-5:** Students discuss design choices, compare preferences, ask questions, and seek input from group members with diverse abilities, experiences, and perspectives.
3. **Create and Accept Feedback:** • **K-2**: With teacher scaffolding, students seek help and share ideas to achieve a particular purpose. Students ask questions of others and listen to their opinions. • **3-5:** Students provide and receive feedback related to computing in constructive ways. For example, pair programming is a collaborative process that promotes giving and receiving feedback.
4. **Use Collaboration Tools:** • **K-2:** Students collaboratively brainstorm by writing on a whiteboard or paper. • **3-5:** Students use collaboration tools to manage teamwork and utilize online project spaces. They also begin to make decisions about which tools would be best to use and when to use them.
Instructional Considerations for Collaboration Practices

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

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

Instructional activities may include but are not limited to:

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

B. **Fostering Computational Thinking Practices**
1. **Decompose Real-World Problems:** • **K-2:** Students break problems, information, and processes into parts. Identify relationships and connections among parts. Reflect on how decomposition aids problem-solving across contexts. • **3-5:** Students further break problems into subproblems, apply systems thinking to explore interdisciplinary connections and integrate existing solutions or procedures (i.e. classroom processes, math procedures, school routines) Apply algorithms to break a problem into subtasks that can be solved and combined to solve the main problem.
2. **Explore Common Features and Identify Patterns:** • **K-2:** Students will be able to identify and describe repeated sequences in data or code through analogy to visual patterns or physical sequences of objects. Students will identify patterns, such as recognizing repeated patterns of code that could be more efficiently implemented as a loop. • **3-5:** Students analyze patterns to develop generalizations and models, test their limits, and validate inputs. Use patterns to analyze trends, justify design decisions, and create artifacts.
3. **Use Abstraction to Simplify, Represent, and Problem Solve:** • **K-2:** Students use and/or create abstractions (e.g., storyboards, flowcharts, decision trees, models) to simplify problems, represent information, organize thinking, communicate, and create artifacts. Artifacts can include but are not limited to prototypes, programs, planning documents, and animations. • **3-5:** Students use and/or create abstractions (e.g., visualizations and computer simulations) to simplify problems, represent information, organize thinking, communicate, and create artifacts. Students intentionally use abstractions to support the problem-solving process to aid in understanding, planning, and predictions.
4. **Apply Algorithmic Thinking to Problem Solve and Create:** • **K-2:** Students use algorithmic thinking to develop a sequence of steps to plan, create, test, and refine artifacts with and without technology. • **3-5:** Students use pseudocode and generalizations to organize, create and seek and incorporate feedback on more complex designs.
5. **Apply Computational Thinking Practices to Select, Organize, and Interpret Data:** • **K-2:** Students use computational thinking to organize data and make predictions. Explore parts and relationships within data sets. • **3-5:** Students visualize data. Use patterns and algorithmic thinking to organize data, identify trends, and make predictions. Use decomposition to explore parts and relationships within data sets. Ask questions about available data sources, and compare and analyze test results to inform decisions, plan, and refine designs.

Instructional Considerations for Computational Thinking Practices

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

1. Model strategies for breaking complex information into smaller parts. Provide opportunities to analyze and discuss the relationship among parts.
2. Support students with recognizing patterns. Model how to analyze, interpret, and display patterns to make predictions and draw conclusions.
3. Model the use of abstraction (e.g., visualizations, storyboards, flowcharts, decision trees, models, computer simulations) to simplify problems; represent information (e.g., data, patterns, processes, phenomena, systems); organize thinking; and support sense-making. Support students with creating and evaluating abstractions and their limitations.
4. Plan opportunities for students to use sequencing in problem solving, incorporate user feedback, and check for bias, accessibility, and other design criteria. Model ways to systematically test, validate, evaluate, refine, and optimize algorithmic solutions. Provide opportunities to reflect on how algorithms are used in solutions.
5. Model abstraction, pattern analysis, and decomposition. Use models to develop and test predictions. Identify limitations and benefits of models.

Instructional activities may include but are not limited to:

- **Create an Artifact**: Students could create an app, program, animation, simulations, etc. to solve a community problem or creatively express an idea.
- **Identify Patterns to Make Predictions**: Students notice repetition in sequences of numbers or parts of a process to make predictions about future events or missing components.
- **Create Abstractions:** Students choose the best tool to use for problem solving using abstractions. Discuss which tools worked best for the team and the problem. Tools may include models, visualizations, storyboards, flowcharts, decision trees, generalizations, simulations.
- **Create Models:** Develop models to represent information such as patterns, relationships, inputs/outputs. Create models of systems (e.g. model networks, cybersecurity, emerging technologies) to understand how parts connect to perform a function. Students can create models to engage in systems thinking and modularization.
- **Evaluate existing models and programs:** Evaluate outputs for bias, accessibility, reliability or other established design criteria. Students can identify applicable parts or modules of existing programs and reuse to solve different problems.
- **Reflection and Transfer:** Have students reflect on how each computational practice facilitates problem-solving and identify opportunities to apply the practice to other situations. Support students identify key points in feedback.

C. Fostering Iterative Design Practices

1. **Identify, Define, and Evaluate Real-world Problems:**
   - **K-2:** With guidance from an educator, identify, define, and explore existing problems and potential solutions. Ask questions to view problems from different perspectives.
   - **3-5:** Students identify, define, and explore existing problems and potential solutions. Ask questions to understand problems from different perspectives. Clarify success criteria, identify constraints, and uncover missing information. Explore patterns and develop generalizations about the types of problems that benefit from computational solutions.

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

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

4. **Test and Optimize Artifacts:**
   - **K-2:** Students test artifacts to ensure they meet criteria and constraints, comparing results to intended outcomes. Use computational thinking and other problem-solving strategies like trial and error to fix simple errors, debug, revise, and evaluate artifacts against design criteria.
   - **3-5:** Students test artifacts to ensure they meet criteria and constraints, comparing results to intended outcomes. Use computational thinking and other problem-solving strategies like trial and error to fix simple errors, debug, revise, and evaluate artifacts against design criteria. Reflect on how the iterative design and computational thinking practices facilitate program development.

Instructional Considerations for Iterative Design Practices

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

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

Instructional activities may include but are not limited to:

- **Class Discussions:** Discuss 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 implementing 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 identify and digital footprint. Incorporate learning activities like discussions of digital dilemmas that help students explore different perspectives, benefits, risks, and tradeoffs.
2. Incorporate opportunities for students to reflect on possible positive and negative impacts of how they use computing technologies. Choose instructional technology that aligns with learning goals and use data on students learning to reflect on and assess the extent

to which the technology is supporting learning outcomes. Provide opportunities to identify the role of humans in developing and using technology.

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 2 Computer Science Vocabulary

Vocabulary Word	Definition
Abstraction	A filtering process used to create a simplified representation of relevant data to identify essential details, excluding less important details.
Algorithm	Finite specified, set of step-by-step instructions designed to solve a problem or perform a task.
Algorithmic Thinking	Process of developing algorithms in a logical, systematic, and procedural way to solve problems or complete tasks.
Acceptable Use Policy (AUP)	Rules and guidelines that define safe practices and responsible use of technology.
Authentication	A process used to verify a user's identity before accessing a network or computer system.
Author	The creator of a book, image, song, or object.
Binary	A number system that uses two digits, 0 and 1.
Block-Based Programming	A visual drag and drop programming tool that users can use to create programs using command blocks.
Categorical Data	Are observations about characteristics that can be sorted into groups or categories (e.g., qualitative). It may include photos, text images, videos, non-numerical values, survey responses, true/false values, or colors.
Character	A person or animal in a book, story, movie, or project. A letter, number, or symbol used in a password.
Code	Any set of instructions expressed in a programming language.
Collecting	Gathering the appropriate type of data needed.
Computational Artifacts	Any creation made by a human using a computing device. It 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).
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 Visualization	The representation of data through use of common graphics, such as charts, plots, infographics and even animations to make complex data more accessible and understandable.
Debug	Process of identifying, isolating, and fixing errors (often referred to as "bugs") in a set of instructions, code, or system. This can also include hardware and software.
Decomposition	Process of breaking down a problem, process, or task into smaller, more manageable components.
Design	Creation of a plan or prototype of a proposed solution.
Design Document	A detailed plan that outlines the structure, features, and implementation strategy of a project. It serves as a blueprint, providing clear specifications, goals, and guidelines for developers, designers, and stakeholders. Design documents often include diagrams, technical requirements, workflows, and rationale to ensure a shared understanding of the project's direction and execution.
Diagrams	Visual representation of data, information, or concepts.
Email	A program used to send and receive messages over the Internet for online communication.
Encode	Convert (information or an instruction) into a particular form.
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., mouse clicks on the run block.

Flowchart	A diagram that shows the steps in a process using shapes and arrows. It helps the user visualize how things happen in order.
Hardware	The physical components of a computing device that you can touch, such as the processor, memory, keyboard, and display.
Healthy Screen Habits	Practices that emphasize balanced use of digital devices to support physical, mental, and emotional well-being.
Implementation	The development or execution of a functional prototype, program, or product.
Information	Facts provided or learned about something or someone.
Intellectual Property	A person's own creations of the mind, such as inventions, drawings, stories, and poems.
Internet	A global network of interconnected computing devices that allows devices to share information and resources.
Iteration	Repeated actions.
Iterative Design Process	A systematic approach to creating and refining products, systems, or solutions through repeated cycles of design, evaluation, and improvement.
Key	A distinct identifier used to differentiate data elements within a set.
List	A data structure that stores an ordered collection of elements, which can be of any type (numbers, strings, objects, etc.)
Loop	A set of instructions that are repeated until a specified condition is met, or a predetermined number of repetitions has occurred.
Memory	Physical storage in computing devices where data is processed and instructions for processing are stored. Memory types include RAM (Random Access Memory), ROM (Read-Only Memory), and secondary storage like hard drives, removable drives, and cloud storage.
Model	A simplified representation of an idea, object, system, or process that helps describe, test, or predict how something works often using diagrams, simulations, or code.
Networking	A group of computing devices (personal computers, phones, servers, switches, routers, etc.) connected by cables or wireless media for the exchange of information and resources.
Numerical Data	Are values or observations that can be measured (e.g., quantitative). It may include heights, temperatures, scores or grades, or statistics.

Output	Data or information produced by a computing device after processing input.
Output Devices	Hardware components that display processed data or information.
Password	A secret word or phrase used to protect devices and information from unauthorized access.
Pattern Analysis	Process of identifying commonalities, differences, and predictable relationships within data to understand, interpret, and make predictions.
Pattern Recognition	Ability to identify commonalities, similarities, or differences in recurring elements.
Personal Information	Data or information about a person that relates to their identity, characteristics, or activities.
Plain Language	A description of the steps and logic in simple terms that anyone can understand through the use of familiar analogies, real-life examples, and simple terms.
Private Information	Sensitive data or information that can identify a person or give others access to your personal life.
Problem Definition	Clearly identifying the problem or challenge that needs to be solved.
Procedure	Broadly used to refer to a process, which may include a method, function, subroutine, or module, depending on the programming language.
Program	The implementation of an algorithm (set of instructions) translated into a programming language that a computer can follow and execute to perform a specific task.
Programming Language	A structured system for writing instructions that a computer can understand and execute. It includes syntax, which defines the rules for how code is written, and semantics, which conveys the meaning of the instructions. Programming languages enable developers to build software, automate processes, and control computer hardware.
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.
Screen Time	Time spent on a computing device.
Selection	Using conditions to manage the sequence of a program's execution.
Sequence	The specific order in which instructions or steps are executed in an algorithm or program.
Simulation	Replicating the behavior of a real-world process or system over a period of time.
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.

Syntax	Rules or structure of a programming language.
Table	A structured format to organize and record information in rows and columns.
Testing	The process of evaluating a program or system to assess its results and outputs, ensuring accuracy, performance, and reliability, while identifying errors.
Trends	Are long-term directions or movements in data or behavior that indicate a general tendency or shift over time.
Troubleshoot	Processes used to diagnose why a system or process is not working as expected and systematically testing solutions to resolve the issue.
Unauthorized Access	Information is accessed without the permission of the owner.
User Input	Data or information that a user provides to a computer program during its execution (2024). Data that is taken in by a computer for processing (2017).
Username	A unique name that people use to log into a device or online account. It is like a nickname that helps the computer recognize who is logging in.
Video Call	A live conversation where people can see and hear another person while talking through a connected device, like a tablet or computer.
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.