2026 CSTA PK–12

COMPUTER SCIENCE STANDARDS

Elementary Implementation Guidance

Introduction

The CSTA PK–12 Computer Science Standards are designed not only to define what students should learn, but also to help the CS education community translate that vision into classroom practice. Educators, curriculum developers, professional learning providers, and state and district leaders all play a role in bringing the Standards to life through rigorous learning experiences that prepare students to understand and shape a world powered by computing.

The Standards were written with practical implementation in mind. They assume that students experience a coherent progression of computer science learning across grade bands. In elementary school (grades PK–5), students engage in approximately 20–40 hours of CS learning per year. In middle school (grades 6–8) and high school (grades 9–12), students experience the equivalent of a yearlong course at each grade band.

These learning opportunities may occur through dedicated computer science courses or through thoughtful integration across subject areas. Because schools and districts vary widely in their schedules, resources, and instructional models, the Standards are designed to support flexible implementation across diverse school contexts while maintaining coherent learning progressions for students.

Purpose of This Guidance

Computer science in PK–5 develops logical reasoning, computational thinking, and problem-solving while fostering creativity, supporting social-emotional growth, and preparing students for a technology-driven world. These foundations matter because elementary school establishes the knowledge, practices, and dispositions students build on throughout the middle and high school progression.

This guidance is designed for instructional leadership teams making decisions about how to implement PK–5 computer science in ways that are coherent, realistic, and aligned to the standards. Because elementary schools vary widely in staffing, scheduling, and specialist availability, there is no single answer to how CS should be delivered. This document presents three models to do so: integration into the general education classroom, integration into an existing special class, and a standalone CS class. The goal is that computer science instruction reaches all students; exactly how it is implemented should be flexible and based on local needs and available resources. Throughout planning, teams should draw on the implementation examples and interdisciplinary connections included with each standard, which illustrate what instruction can look like across a variety of classroom contexts without prescribing specific curriculum.

For easier viewing and more details, check out the interactive web display.

Why Begin Computer Science in Pre–Kindergarten

Early childhood is a foundational period for developing the problem-solving, logical reasoning, and creative thinking skills essential to computer science. Many of these skills are already present in a developmentally appropriate early learning setting grounded in hands-on problem solving, an approach aligned with best practices in early childhood education that lets children "play to learn while learning to play" (Resnick, 2003).

In PK and Kindergarten, these skills can be introduced and nurtured through unplugged activities: hands-on, child-centered learning that does not require screens or digital devices. By engaging in interactive games, movement, storytelling, and manipulatives, young learners begin to think computationally, sequencing steps, recognizing patterns, and solving problems in ways that are developmentally appropriate and accessible to all. Beginning in PK ensures every student enters the elementary progression with this foundation rather than encountering computational thinking for the first time in later grades.

Artificial Intelligence in Elementary Computer Science

Artificial intelligence is a field of study within computer science, and even the youngest students should begin to learn how AI technologies work at grade-appropriate levels as defined in the standards. Elementary students examine how AI technologies use data to make decisions, how those technologies affect their lives and society, and the differences between what people and AI technologies do well.

Elementary students must realize that AI technologies are not magic and should not be treated as human. Educators should avoid anthropomorphizing AI so that students build realistic mental models of what AI can and cannot do. (Learn more in the Raspberry Pi Foundation's Pedagogy Quick Read: The Effects of Anthropomorphisation on Students' Mental Models of AI).

The focus of AI in elementary computer science is not teaching students to use generative AI tools; it is helping them build knowledge of how AI technology works so they can make informed decisions about their use of AI in a world powered by computing.

Why Define Standards for Each Grade Level

Previous CSTA standards and a majority of state-adopted CS standards define grade-band standards. In contrast, the 2026 CSTA PK–12 Standards define specific grade-level expectations for several reasons. This is consistent with other national subject area standards, such as Common Core and NGSS.

1. Adds Clarity and Precision in Expectations

Grade-level standards provide clear, specific learning objectives, making it easier for teachers to know exactly what students are expected to achieve at each stage. This reduces ambiguity compared to grade bands, where expectations are spread over several years and can lead to uneven progression.

2. Improves Curriculum Alignment and Instruction

With grade-level standards, curriculum developers and educators can design lessons and assessments that directly align with yearly goals and, if desired, allow for better integration with other subject areas. This ensures a more coherent and consistent learning experience for students, preventing gaps or unnecessary repetition in instruction.

3. Enhances Student Progress Monitoring

Grade-level standards make it easier to track individual student progress annually. Educators can more easily identify misconceptions and identify students who are excelling or struggling and provide timely interventions. This is more challenging with broader grade bands.

4. Ensures More Equitable Access to Learning

When standards are set by grade level, all students are guaranteed exposure to the same content at the same time, regardless of school or district. Grade bands may lead to some students missing key concepts if schools choose to focus on different areas within the banded years.

5. Streamlines Transition Between Grades

Grade-level standards support smoother transitions from one grade to the next, as teachers know they can build directly on the previous year's learning. This continuity is harder to maintain with banded standards, where teachers may not know exactly what was covered in earlier grades.

Organizing computer science standards by grade level increases clarity, consistency, and equity in student learning, making it easier for educators to deliver high-quality computer science education. This allows students to build skills year by year, leading to an increase in confidence and interest in computer science as they advance through their education.

Design Principles for Elementary School

Principle	Description
Equitable Access	Elementary CS should not be reserved for a small group of students or treated as optional enrichment for those already interested in technology. Decisions should be made with participation in mind: who has access, who is being left out, and what policies and schedules are needed to ensure every student has meaningful opportunities to learn.
Coherent Learning	The standards are a progression. The PK and Kindergarten standards are grouped in a grade band, so skills introduced in PK are reinforced in Kindergarten, and elementary implementation should extend this logic across the full band toward readiness for middle school. Even when schools deliver CS differently, learning should be built intentionally rather than repeating disconnected introductory experiences.
Flexibility	Strong implementation can look different across schools and districts. This document does not present a single required model; it offers implementation models schools can combine to fit their own context while maintaining the integrity of the standards.
Alignment to Intent	Computer science is a foundational way of thinking that overlaps with what teachers already teach. Whether students define sorting rules for a graph or sequence events in a narrative, the CS element is the logical rules, structured data, and analysis they apply, with or without devices. Implementation should reflect this full intent rather than reduce CS to screen time.
Evidence-Based Pedagogy	Research-based, inclusive pedagogy is fundamental to a classroom where every student feels valued and supported. Teachers should build understanding before independent creation, make abstract ideas concrete through hands-on tasks, and treat debugging and revision as normal parts of learning. Instruction should reduce barriers using the Universal Design for Learning Guidelines for Computer Science Education, with practical strategies available through the Raspberry Pi Foundation's Pedagogy Quick Reads.
Attention to Local Capacity	Implementation decisions should be shaped by who will teach the content, when students will experience it, and what support educators need, alongside what schools hope to offer. Elementary CS implementation is a question of staffing, instructional time, PD, and the systems needed to support consistent, high-quality learning, in addition to curriculum.

Key Decisions Schools Need to Make

Successful implementation requires planning about how CS will function within the elementary program as a whole. The questions below help schools assess current practice and identify the adjustments needed for alignment, coherence, and equitable access under the elementary standards. Schools may find it useful to revisit these questions as implementation evolves.

1. **How do students currently experience computer science across PK–5?**

   Before making changes, schools should examine:
   - Where students already encounter computer science learning
   - Whether access is consistent across grade levels and student groups
   - Which experiences reach all students versus only some classrooms

2. **Who will teach computer science, and how will they be supported?**

   Elementary delivery depends heavily on who is responsible for instruction and whether they feel prepared. Schools should consider:
   - Whether CS is taught by classroom teachers, specialists, or both
   - What content knowledge and PD will those educators need
   - What planning time and instructional support are available

3. **How will computer science fit into the schedule?**

   Because elementary schedules are already full, schools should plan for:
   - Whether CS is integrated into existing subjects, delivered in a special class, or both
   - How much dedicated instructional time will students receive
   - How CS time is protected from being cut or absorbed by pull-out programming

4. **How will coherence and progression be maintained?**

   Because elementary CS builds from the earliest grades and feeds into middle school, schools should plan for:
   - Intentional sequencing of standards and practices across PK–5
   - Shared expectations for depth and complexity at each grade
   - Coordination across classrooms, specialists, and grade levels

The following reflection framework is designed to help leadership teams take stock of what is currently happening across PK–5 and identify the most important shifts needed for standards-aligned implementation. Teams can complete it collaboratively, revisiting it as scheduling, staffing, and curriculum decisions evolve.

Current State → Future State Reflection

Reflection Question	Current State	Where We Want to Go
Where do students encounter CS?		
Who has access?		
Where do concepts repeat without growth?		
Where are standards not yet addressed?		

Once you have considered the framework above, use the matrix below to compare the three implementation models across key decision areas. The models describe ways to organize instruction, not mandates to replace existing courses, and many schools will combine elements of more than one. The matrix can support clearer conversations about tradeoffs and help schools select an approach suited to their context.

Key Decisions × Implementation Models Matrix

Decision Area	Model 1: Integrated into General Education	Model 2: Integrated into a Special Class	Model 3: Standalone CS Class
Student Exposure	Every student, within their classroom	All students who take the special class	All students are scheduled into the class
Structure	Embedded across daily instruction	Embedded in STEAM, media, or makerspace	Dedicated CS time, scheduled like music or art
Staffing	Classroom teachers	Existing specialists (e.g., STEAM, library media)	Dedicated CS specialist
Coherence	Cross-subject planning by grade	Coordination between specialist and classroom	Vertical CS planning across grades
Best Fit	Schools building CS capacity within existing staff	Schools with established special classes	Schools able to dedicate staffing and time

Each of these models is explored in depth in the sections that follow, including classroom examples and planning considerations.

Model 1: Integrated into the General Education Classroom

The classroom teacher embeds computer science into the subjects they already teach, so students encounter CS as part of math, science, ELA, and social studies rather than as a separate subject. This is often the most accessible starting place for elementary schools, since it reaches every student without depending on specialist staffing or a separate schedule.

Integration works because computer science overlaps naturally with what elementary teachers already do. When students sort objects by attribute, sequence a story's events, or collect and represent data, they are already practicing computational thinking. The teacher's role is to make that learning explicit, drawing out the algorithms and reasoning embedded in existing lessons rather than adding disconnected activities to a full day.

Integration can look like just-in-time teaching as CS concepts arise, a regular block of dedicated CS time, or standards mapped across real-world projects. The examples that follow show three points along that range, from a no-screens, play-based approach in pre–kindergarten to project-based integration in grade 4.

Is This Model Right for Your School?

This model fits schools that want all students to experience CS without adding specialist staff or restructuring the schedule, and that can support classroom teachers in building CS content knowledge.

- **Main strength – reach:** CS lives inside the regular classroom, reaching every student and connecting computing to authentic, cross-disciplinary contexts.
- **Main challenge – teacher support:** Integration depends on teachers understanding the standards well enough to make CS learning explicit, which requires PD and planning time so CS is genuinely taught rather than assumed.

Schools using this model should treat CS as intentional, standards-aligned learning, not incidental technology use, and give teachers the preparation that requires.

Planning Resources

Three worked examples support planning in this model, together showing how general education integration adapts across the PK–5 range:

- **Pre–Kindergarten examples** – Shows a no-screens, play-based approach to integration.
- **Grade 2 examples** – Shows integration at a mid-elementary point in the band.
- **Grade 4 examples** – Shows project-based integration as students take on more complex tasks.

Pre-Kindergarten: Integrating CS through Play

Classroom Snapshot: A Pre-Kindergarten teacher leads instruction using hands-on materials (craft materials, building materials) and CS-related books, emphasizing problem solving, computational thinking, and storytelling through a no-screens, hands-on, active approach with just-in-time teaching of CS concepts as needed.

Mathematics & Science: **Sorting and Counting Nature Items**

Subject Standards	• ELOF.P-MATH 1: Child knows number names and the count sequence. • ELOF.P-MATH 2: Child recognizes the number of objects in a small set. • ELOF.P-MATH 5: Child associates a quantity with written numerals up to 5 and begins to write numbers. • ELOF.P-SCI 3: Child compares and categorizes observable phenomena.
CS Standards	• EK-ALG-ML-02: Recognize attributes of objects to notice patterns and make decisions. • EK-DAT-DC-08: Use collected data to help answer questions. • EK-DAT-DI-09: Investigate a question that can be answered by collecting data in students' everyday environments.
Math & Science Activity	Children sort a collection of nature items (e.g., leaves, rocks, sticks) by attribute, then count and record how many items are in each category (e.g., tally marks, number cards).
Math & Science Activity with CS Integration	To answer the question "How many different colors of natural items can we find?", children collect nature items on a walk, sort and count them by color, then record their counts on a class chart to determine which color was most common.

English Language Arts: **Story sequencing**

Subject Standards	• ELOF.P-LIT 4: Child demonstrates an understanding of narrative structure through storytelling and re-telling. • ELOF.P-LC 5: Child expresses self in increasingly long, detailed, and sophisticated ways.
CS Standards	• EK-ALG-PS-01: Carry out algorithms in daily activities. • EK-PRO-PD-04: Create a sequence of commands to solve a problem or express an idea. • EK-PRO-TR-07: Identify a step in a sequence of commands that does not work as expected.
ELA Activity	Children use puppets to act out and retell events from a familiar story.
ELA Activity with CS Integration	To retell a story, students sequence picture cards using transition words (first, next, last), then identify and rearrange any cards that are out of order before acting out the sequence with puppets.

English Language Arts: **Computer Scientist Read-Aloud**

Subject Standards	• ELOF.P-LIT 5: Child asks and answers questions about a book that was read aloud. • ELOF.P-LC 2: Child understands and responds to increasingly complex communication and language from others.
CS Standards	• EK-SOC-HU-15: Explain that people design and develop computing technologies. • EK-SOC-CE-16: Identify how people use digital devices at home, at school, and at work.
ELA Activity	The teacher conducts an interactive read-aloud, thinking aloud and pausing to ask questions, and children draw and write or dictate a sentence about a key point.
ELA Activity with CS Integration	The teacher selects a read-aloud about a computer scientist, a community helper who uses CS, or how people use digital devices, directing questions toward how people design and use technology. Children draw a picture and write or dictate a sentence about what it shows.

Approaches to Learning (AtL) & Mathematics: **Pattern Making**

Subject Standards	• ELOF.P-MATH 7: Child understands simple patterns. • ELOF.P-ATL 11: Child shows interest in and curiosity about the world around them.
CS Standards	• EK-ALG-PS-01: Carry out algorithms in daily activities. • EK-PRO-RD-06: Describe how a sequence of commands completes a task.
AtL Activity	Children create a repeating pattern using stamps, stickers, blocks, or other materials.
AtL Activity with CS Integration	After creating a pattern, children extend a friend's pattern as a sequence, then explain how their extension follows the same rule.

Approaches to Learning (AtL): **Classroom Rules Sorting**

Subject Standards	• ELOF.P-ATL 2: Child follows classroom rules and routines with increasing independence. • ELOF.P-ATL 3: Child appropriately handles and takes care of classroom materials. • ELOF.P-LC 2: Child understands and responds to increasingly complex communication and language from others.
CS Standards	• EK-SYS-IM-13: Identify responsible behavior when using computing systems and digital tools. • EK-PRO-VD-05: Identify everyday gestures and symbols that represent information people use to make choices.
AtL Activity	During circle time, children use gestures (e.g., thumbs up/down) to sort pictures of classroom behavior into Do and Don't categories.
AtL Activity with CS Integration	During circle time, children use gestures (e.g., thumbs up/down) to indicate whether a picture shows responsible or irresponsible use of a computing device (e.g., handling a tablet gently vs. eating near one), then sort the pictures into Do and Don't categories.

Grade 2: Integrating Computer Science Plus Dedicated CS Instruction

Classroom Snapshot: A 2nd grade classroom teacher leads instruction using robots, devices to program robots and run programming apps, and classroom/craft/building materials, with about 10 hours of dedicated CS teaching (about 30 minutes every other week), plus application of CS concepts across curricular areas and using CS vocabulary daily.

Integrated CS Lessons

Mathematics: **Data Collection & Visualizing**

Subject Standards	• CCSS.MATH.CONTENT.2.MD.D.10: Draw a picture graph and a bar graph (with single-unit scale) to represent a data set with up to four categories. Solve simple put-together, take-apart, and compare problems using information presented in a bar graph.
CS Standards	• E2-DAT-DC-08: Compare numeric and non-numeric types of data in terms of how they are collected and what information they provide. • E2-DAT-DI-09: Develop a question that can be answered with data. • E2-DAT-IM-10: Distinguish between data collection approaches, including those that may lead to inaccurate or biased data.
Math Activity	Students work in pairs to poll classmates about favorite lunch foods using a tally sheet, record findings on a class picture graph, and answer comprehension questions about the graph.
Math Activity with CS Integration	Students propose and vote on a data question (e.g., favorite sport), then poll an assigned group of students in the school, using the first four unique answers as categories. They create a labeled bar graph, identify the favorite and least favorite categories, and discuss why different groups' results differ and how their collection method could be improved.

English Language Arts: **Storyboard Narrative**

Subject Standards	• CCSS.ELA-Literacy.W.1.3: Write narratives in which they recount two or more appropriately sequenced events, include some details regarding what happened, use temporal words to signal event order, and provide some sense of closure. • CCSS.ELA-Literacy.W.1.6: With guidance and support from adults, use a variety of digital tools to produce and publish writing, including in collaboration with peers.
CS Standards	• E2-ALG-PS-01: Create an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-PRO-PD-04: Create code from an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-PRO-TR-07: Debug a program that includes sequence, events, and iteration.
ELA Activity	Students use a storyboard template to plan and write a story about a personal event, using temporal words to sequence it.
ELA Activity with CS Integration	Students illustrate their personal-event story as an algorithm with correctly sequenced events, using a template, then use the algorithm to create a programmed animation of the story and test and fix it until it runs correctly.

English Language Arts: **Mad Libs**

Subject Standards	• CCSS.ELA-Literacy.L.2.1: Demonstrate command of the conventions of standard English grammar and usage when writing or speaking. • CCSS.ELA-Literacy.L.2.1.e: Use adjectives and adverbs, and choose between them depending on what is to be modified.
CS Standards	• E2-PRO-VD-05: Label different representations of information with a name and whether its value is constant or changes.
ELA Activity	As a class, students identify parts of speech to fill blanks in a Mad Libs story, then complete a second story in small groups, taking turns supplying words.
ELA Activity with CS Integration	Students label each blank of a Mad Libs story with a name describing its part of speech (e.g., noun, adjective) before filling it in, noting that some elements, like the story's title, stay constant while others change.

Social Studies: **Historical Changes**

Subject Standards	• D2.His.3.K-2: Generate questions about individuals and groups who have shaped a significant historical change. • D2.His.14.K-2: Generate possible reasons for an event or development in the past. • D2.His.16.K-2: Select which reasons might be more likely than others to explain a historical event or development.
CS Standards	• E2-SOC-HI-14: Analyze the ways that people from different cultures, backgrounds, and time periods have designed computing technologies to help them solve problems and express ideas. • E2-SOC-HU-15: Investigate situations where humans have created computing technologies to solve problems.
Social Studies Activity	Students brainstorm questions about a historical figure, then in small groups research and evaluate possible reasons behind a related event using teacher-provided materials, discussing whether their evidence answers their original questions
Social Studies Activity with CS Integration	Students brainstorm questions about a person who designed a computing technology (e.g., Ada Lovelace, Tim Berners-Lee, Steve Wozniak, Mark Dean, Fei-Fei Li), then research and evaluate possible reasons behind the problem that person's work solved or the idea it expressed.

Social Studies & ELA: **Career Development**

Subject Standards	• D2.Eco.6.K-2: Explain how people earn income. • CCSS.ELA-Literacy.W.2.3: Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.
CS Standards	• E2-SOC-CE-16: Investigate how personal interests connect to computing in different industries and careers.
Social Studies & ELA Activity	Guest speakers from various industries visit for Career Day; students ask about their work and needed skills, record answers on a template, then write thank-you notes.
Social Studies & ELA Activity with CS Integration	After a class survey of student interests, guest speakers matched to those interests describe at least one computing technology they use in their job. Students record how computing applies to each speaker's work, build a class chart linking interests to careers, draw themselves in a future career with computing applications, and write thank-you notes.

Science: **Landforms and Bodies of Water** – Two Versions: 1) Robot Navigation, 2) Machine Learning Classification

Subject Standards	• NGSS.2-ESS2-2: Develop a model to represent the shapes and kinds of land and bodies of water in an area. • NGSS.2-ESS2-3: Obtain information to identify where water is found on Earth and that it can be solid or liquid.
Science Activity	Students identify landforms and bodies of water by observation, video, or text, then create and compare a 3D model of a land and water area with a partner.
CS Standards (1)	• E2-ALG-PS-01: Create an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-PRO-PD-04: Create code from an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-PRO-TR-07: Debug a program that includes sequence, events, and iteration.
Science Activity with CS Integration (1)	Students work in pairs to create, test, and improve an algorithm that programs a robot to reach locations of water on a floor map, using sound, lights, or actions to show whether the water is liquid or solid.
CS Standards (2)	• E2-ALG-ML-02: Examine how data is used to train a machine learning model.
Science Activity with CS Integration (2)	Students sort images of the landforms and bodies of water into categories to train a machine learning model. The class tests the model on new images, discusses its accuracy, and suggests changes to the training data if it misclassifies.

Dedicated CS Lessons

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Systems & Security (4 hours)	What is a computer and how do they impact us?	Students read books/articles or watch videos about input and output, then create an illustration or physical model of a computing device labeling input and output components, and produce a video describing how those components work. They role-play scenarios about how online actions (e.g., cyberbullying, using someone else's password) can lead to repercussions, and discuss how they, their friends, and family use computing systems, identifying benefits and harms.	• E2-SYS-HW-11: Explain how the basic hardware components of a computing system work together to perform input and output (I/O) operations. • E2-SYS-SE-12: Explain how online actions have real-world consequences. • E2-SYS-IM-13: Describe the benefits and harms that arise from an individual's use of computing systems and digital tools.	SEL (online safety, real-world consequences of actions) ELA (descriptive writing for video narration)
Algorithms & Design and Programming (6 hours)	How can we use code to turn a task into an automated solution?	Unplugged: in pairs, one student acts as a programmer writing an algorithm with coding cards for the other to execute as a robot; the class follows a teacher's lining-up algorithm and discusses who it might and might not work well for. Coding App: students illustrate a dance sequence as an algorithm using a template, translate it into a working program, and describe their code to a peer. Robot: in pairs, students program a robot to navigate between locations and to trigger an event based on its sensors, then identify the sensors used and explain the code involved.	• E2-ALG-PS-01: Create an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-ALG-IM-03: Describe how an algorithm might impact peers in varied situations. • E2-PRO-PD-04: Create code from an algorithm that includes sequence, events, and iteration to solve a problem or express an idea. • E2-PRO-RD-06: Explain the steps taken to create a program. • E2-PRO-TR-07: Debug a program that includes sequence, events, and iteration.	Arts/Dance (dance sequence as algorithm) Math (spatial directions for robot navigation)

Grade 4: Integrating Computer Science Using Project-Based Learning

Classroom Snapshot: A 4th grade classroom teacher leads instruction using classroom materials and manipulatives and devices with block-based coding apps, intentionally integrating computer science into real-world projects throughout the curriculum.

Earth Science: **Data Collection & Visualizing**

Subject Standards	• NGSS 4-ESS2-2: Analyze and interpret data from maps to describe patterns of Earth's features and weather. • CCSS.MATH.CONTENT.4.MD.B.4: Make a line plot to display a data set of measurements in fractions of a unit (1/2, 1/4, 1/8).
CS Standards	• E4-DAT-DC-09: Organize collected data into a table using a computational tool, with rows representing records and columns representing attributes. • E4-DAT-DI-10: Create an explanation that includes at least one data visualization to report the process and results of a data investigation. • E4-ALG-ML-02: Analyze relationships between the properties of training data and a machine learning model's output. • E4-SOC-HU-18: Distinguish between human learning and machine learning processes.
Earth Science Activity	Students measure water levels in physical rain gauges representing different cities to the nearest 1/4, 1/2, or 1/8 inch and record their fractional measurements in a science journal.
Earth Science Activity with CS Integration	Students organize their rainfall measurements into a digital spreadsheet by city (rows) and rainfall amount (columns), then create a line plot showing rainfall frequency. They compare how a computer could use the dataset to classify regions as heavier or lighter precipitation with how people recognize patterns in examples.

Science: **Energy, Devices, and the Environment**

Subject Standards	• NGSS 4-ESS3-1: Obtain and combine information to describe that energy and fuels are derived from natural resources and their uses affect the environment.
CS Standards	• E4-SYS-IM-15: Investigate the impacts of widely used computing systems on natural resources and the environment. • E4-SYS-HW-12: Apply a basic troubleshooting process to identify and fix common hardware and software issues.
Science Activity	Students use classroom devices to research how electricity, batteries, solar power, and fossil fuels are used in daily life and discuss their environmental impact.
Science Activity with CS Integration	Students examine the energy sources their computing devices use and discuss how device use affects the environment through energy consumption and electronic waste. They also identify common problems that interrupt device use, such as a low battery or unresponsive app, and describe basic troubleshooting steps.

History: **Industrial Revolution or Westward Expansion & Technology**

Subject Standards	• D2.His.3.3-5: Generate questions about individuals and groups who have shaped significant historical changes and continuities.
CS Standards	• E4-SOC-ET-17: Analyze how the limitations of existing technologies can lead to emerging technologies. • E4-SOC-HI-16: Investigate the contributions of diverse individuals and communities in the history of computing. • E4-SOC-CE-19: Investigate how the workforce adopts new computing technologies and continues to update their computing skills.
History Activity	Students read a non-fiction text about individuals or groups connected to a major historical change (e.g., Industrial Revolution inventors, westward expansion), generate questions about their influence, and discuss how new tools and inventions changed the way people lived and worked.
History Activity with CS Integration	Students investigate a historical computing innovator, diverse group, or community, generating questions about the problem they solved and how their contributions influenced later technologies. They discuss how newer technologies, including AI, build on earlier innovations and how workers keep learning new skills as technology changes, then organize their ideas in a digital timeline, slide, or chart.

ELA: **Biographies & Narrative Writing** – Two Versions: 1) Block-Based Coding 2) Unplugged Sequencing Cards

Subject Standards	• CCSS.ELA-Literacy.RL.4.3: Describe in depth a character, setting, or event in a story or drama, drawing on specific details in the text (e.g., a character's thoughts, words, or actions).
ELA Activity	Students write a paragraph describing a character's traits, a short response describing the setting and its influence on events, and a short narrative about a significant event from the person's life using dialogue, actions, and setting details.
CS Standards (1)	• E4-ALG-PS-01: Create a written representation of an algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E4-PRO-PD-05: Collaborate with a team by offering a meaningful contribution to creating a program. • E4-PRO-PD-04: Compare different programming solutions to the same problem based on correctness and clarity. • E4-PRO-TR-08: Debug a program incrementally and repeatedly throughout the development process. • E4-PRO-RD-07: Document a program to clarify its functionality. • E4-PRO-VD-06: Trace how data flows and changes variable values in a program.
ELA Activity with CS Integration (1)	Working in a group, students create a written storyboard algorithm for an animated biography scene, including a starting event, sequenced actions, repeated dialogue or movement, and a choice about how a character responds. Using a block-based coding app, they animate the scene, incorporating a variable such as scene number or dialogue choice and tracing how its value changes as they test and revise. Groups compare their programmed versions and discuss which is clearer and more effective, then document their code with comments explaining what each part does.
CS Standards (2)	• E4-ALG-PS-01: Create a written representation of an algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea.
ELA Activity with CS Integration (2)	Using dialogue cards, setting cards, and event cards, students build a sequenced biography scene on paper, then explain how the ordered actions, dialogue, and setting details help communicate the important event from the person's life.

Model 2: Integrated into an Existing Special Class

Some schools already have a class built for hands-on, cross-disciplinary work: a STEAM, makerspace, library media, or art class. This model places computer science there, taught by the specialist who already runs it, so students encounter CS with dedicated time and expertise already in place, without a new course or hire.

Integration works because these classes already share practices and tools with computer science: STEAM and makerspace classes are built around design, building, and iteration; library media around information and sequencing; art around pattern and structure. The specialist draws out the CS already adjacent to their content, such as programming robots or coding geometric art, teaching CS Standards intentionally rather than leaving them implicit.

How much CS a special class can carry depends on how often it meets. A library media class meeting twice a month may dedicate set hours to CS-specific skills while weaving CS into existing activities; a weekly STEAM class may integrate CS into most of its lessons. The examples that follow show this range, from kindergarten library media to Grade 3 STEAM.

Is This Model Right for Your School?

This model fits schools with an existing special class and specialist expertise that want to add CS without a new course or asking classroom teachers to take it on.

- **Main strength – existing structure:** CS lands in a class that already has dedicated time, a specialist, and often the right equipment, reaching every student who takes that class without new staffing or schedule changes.
- **Main challenge – coordination and coverage:** The special class shares its limited time with its own subject and rarely reaches every grade, so schools must coordinate with classroom teachers on which standards it covers and which need a home elsewhere.

Planning Resources

Two worked examples support planning in this model, together showing how special class integration adapts across different specialist settings, meeting frequencies, and grade levels:

- **Kindergarten Library Media examples** – Shows integration into a low-frequency special class (two 30-minute lessons per month), where CS is dedicated to specific hours alongside standards woven into existing activities.
- **Grade 3 STEAM examples** – Shows integration into a high-frequency special class (weekly, with CS built into 75% of lessons), where CS is embedded across most of the curriculum.

Kindergarten: Library Media Integration

Classroom Snapshot: A librarian/media specialist leads instruction using robots, devices to program robots, craft materials, and books, dedicating 8 hours (two 30-minute lessons per month) to CS-specific knowledge and skills while weaving CS into existing library media activities where possible.

English Language Arts: **Story Sequencing**

Subject Standards	• CCSS.ELA-Literacy.RI.K.2: With prompting and support, identify the main topic and retell key details of a text.
CS Standards	• EK-ALG-PS-01: Carry out algorithms in daily activities. • EK-ALG-IM-03: Describe how people create algorithms to solve problems. • EK-PRO-PD-04: Create a sequence of commands to solve a problem or express an idea. • EK-PRO-RD-06: Describe how a sequence of commands completes a task. • EK-PRO-TR-07: Identify a step in a sequence of commands that does not work as expected.
ELA Activity	Students arrange physical picture cards of a story's events in chronological order, then retell the story using transition words like "first," "next," and "last."
ELA Activity with CS Integration	Working in pairs, students program a floor robot to navigate to each story event card in order, testing, debugging, and revising their sequence until the robot completes the retelling correctly, then explain how their step-by-step directions helped the robot complete the task.

English Language Arts: **Sorting & Categorizing Books**

Subject Standards	• CCSS.ELA-Literacy.RL.K.5: Recognize common types of texts.
CS Standards	• EK-ALG-ML-02: Recognize attributes of objects to notice patterns and make decisions. • EK-DAT-DC-08: Use collected data to help answer questions. • EK-DAT-DI-09: Investigate a question that can be answered by collecting data in students' everyday environments.
ELA Activity	Students sort books into fiction and nonfiction categories based on features they notice on the cover and inside pages.
ELA Activity with CS Integration	To answer "Which type of book do we have more of?", students sort books into fiction and nonfiction by observable attributes, represent their counts using picture cards, linking cubes, or a digital graph, then use the graph to answer the question.

English Language Arts: **Technology, Tools, and People**

Subject Standards	• CCSS.ELA-Literacy.RI.K.1: With prompting and support, ask and answer questions about key details in a text.
CS Standards	• EK-SOC-CE-16: Identify how people use digital devices at home, at school, and at work. • EK-SOC-HI-14: Identify computing technologies used in daily life that have changed over time. • EK-SYS-HW-11: Examine the use of tools to accomplish a task or solve a problem for different users. • EK-SYS-IM-13: Identify responsible behavior when using computing systems and digital tools.
ELA Activity	Students discuss tools they use in the library, such as books, headphones, tablets, or a document camera, and how people read and find information at home or school.
ELA Activity with CS Integration	Students compare tools used for reading and learning, describing how different tools help different users, then identify computing technologies that have changed over time (e.g., listening centers, tablets, classroom computers) and discuss responsible ways to use digital devices, such as taking turns and asking permission before use.

Grade 3: STEAM Class Integration

Classroom Snapshot: A STEAM specialist leads instruction using robots, devices to program robots, craft and building materials, and books, designing lessons that align computer science concepts to engineering, math, and art for 18 hours (45 minutes weekly across 32 of 36 weeks, with CS integrated into 75% of the class).

Coverage Note: Security, Networks, Impacts of Computing Systems, and Emerging Technology standards are covered separately, during digital literacy/citizenship lessons taught by a different specialist at the school.

STEAM: **Computing Systems Exploration**

Subject Standards	**Engineering:** • 3-5-ETS1-1: Define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost. **Mathematics:** • CCSS.Math.Content.3.MD.C.6: Measure areas by counting unit squares (square cm, square m, square in, square ft, and improvised units). • CCSS.Math.Content.2.MD.A.1: Measure the length of an object by selecting and using appropriate tools such as rulers, yardsticks, meter sticks, and measuring tapes. **Arts:** • MA:Pr4.1.3: Practice combining varied academic, arts, and media forms and content into unified media artworks, such as animation, music, and dance. • MA.Pr5.1.3c: Exhibit standard use of tools and techniques while constructing media artworks. • VA:Cr2.2.3a: Demonstrate an understanding of the safe and proficient use of materials, tools, and equipment for a variety of artistic processes.
CS Standards	• E3-ALG-PS-01: Create an algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-PD-04: Develop code from a student-created algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-RD-07: Articulate how a specific segment of code contributes to the overall purpose of a program. • E3-SYS-HW-12: Describe the role of software in a computing system to accomplish tasks or solve problems.
STEAM Activity with CS Integration	Students label the input sensors on a diagram of their robot, then create code that uses the robot's sensors to take in input, process it, and produce output. They design protective casings and decorative covers that accommodate hardware requirements, then create a video explaining how their software uses the robot's sensors to produce output.

STEAM: **Looping Patterns in Motion and Art**

Subject Standards	**Science:** • 3-LS1-1: Develop models to describe that organisms have unique and diverse life cycles but all have in common birth, growth, reproduction, and death. • 3-LS3-1: Analyze and interpret data to provide evidence that plants and animals have traits inherited from parents and that variation of these traits exists in a group of similar organisms. • 3-ESS2-1: Represent data in tables and graphical displays to describe typical weather conditions expected during a particular season. **Engineering:** • 3-5-ETS1-2: Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem. • 3-5-ETS1-3: Plan and carry out fair tests in which variables are controlled and failure points are considered to identify aspects of a model or prototype that can be improved. **Mathematics:** • CCSS.Math.Content.3.MD.B.3: Draw a scaled picture graph and a scaled bar graph to represent a data set with several categories. **Arts:** • VA:Cr2.1.3a: Create personally satisfying artwork using a variety of artistic processes and materials. • VA:Cr1.2.3a: Apply knowledge of available resources, tools, and technologies to investigate personal ideas through the art-making process. • VA:Re9.1.3a: Evaluate an artwork based on given criteria. • DA:Re.7.1.3a: Find a movement pattern that creates a movement phrase in a dance work. **English Language Arts:** • CCSS.ELA-Literacy.W.3.2: Write informative/explanatory texts to examine a topic and convey ideas and information clearly.
CS Standards	• E3-ALG-PS-01: Create an algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-PD-04: Develop code from a student-created algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-RD-07: Articulate how a specific segment of code contributes to the overall purpose of a program. • E3-PRO-TR-08: Debug a program that includes a combination of sequence, events, iteration, and selection. • E3-ALG-IM-03: Compare how different algorithms may affect outcomes, situations, and people with a wide range of needs. • E3-DAT-DC-09: Evaluate numeric and non-numeric data for accuracy and completeness. • E3-DAT-DI-10: Investigate a data question involving relationships between multiple attributes.
STEAM Activity with CS Integration	Students identify repeating patterns in natural systems (e.g., life cycles, inherited traits, weather conditions) and in dance and visual arts through visual resources. Working with a partner, they program a robot to create geometric artistic compositions using events, sequence, and loops, then graph the efficiency of loops versus a repeated sequence of commands by comparing lines of code and execution time. Students document the process, including explaining how the loops in their program work.

STEAM: **Geometry Navigators**

Subject Standards	**Engineering:** • 3-5-ETS1-1: Define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost. **Math:** • CCSS.Math.Content.3.MD.C.6: Measure areas by counting unit squares (square cm, square m, square in, square ft, and improvised units). • CCSS.Math.Content.2.MD.A.1: Measure the length of an object by selecting and using appropriate tools such as rulers, yardsticks, meter sticks, and measuring tapes. **Art:** • VA:Cr2.2.3a: Demonstrate an understanding of the safe and proficient use of materials, tools, and equipment for a variety of artistic processes.
CS Standards	• E3-ALG-PS-01: Create an algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-PD-04: Develop code from a student-created algorithm that includes a combination of sequence, events, iteration, and selection to solve a problem or express an idea. • E3-PRO-PD-05: Use constructive feedback to improve a program. • E3-PRO-TR-08: Debug a program that includes a combination of sequence, events, iteration, and selection.
STEAM Activity with CS Integration	Students create mazes of specific shapes using a variety of materials, then decorate their robots to represent a vehicle that will navigate the maze. They calculate the distances the robot needs to travel, then work with a partner, giving and incorporating feedback, to program their robot through the maze.

Grade 3: Other Class Integration

The same model applies to art and makerspace classes at Grade 3, though a full worked example isn't developed here. An art specialist could align CS to pattern, structure, and the creative process, following the same logic shown in the STEAM examples above (robots creating geometric art, loops framed as visual repetition). A makerspace class could integrate CS through the design-build-iterate cycle already central to its structure, similar to the Computing Systems Exploration and Geometry Navigators activities. Schools using either class type should map which CS Standards the specialist's content can realistically carry and coordinate with classroom teachers on what remains.

Model 3: Standalone CS Class

In this model, computer science stands on its own: a dedicated CS specialist teaches it as a distinct class with its own time and sequence, scheduled the way elementary schools often schedule music or art. Students receive CS as a subject in its own right rather than something folded into another lesson.

Integration with other subjects still happens in this model, but as enrichment for CS learning rather than the means of delivering it.

A standalone class can take different shapes depending on grade and schedule: a weekly block for part of the year, or a larger allocation that grows across grades. The examples that follow show this progression, from an early dedicated block in Grade 2, to a Grade 1 class organized around short conceptual units, to a Grade 5 class spanning the full standards with deeper, project-based work.

Is This Model Right for Your School?

This model fits schools able to dedicate staffing and schedule time to CS as its own subject.

- **Main strength – depth and coherence:** Protected time and a dedicated specialist let CS be taught directly, sequenced deliberately across grades, and covered to its full breadth rather than only where it overlaps with other subjects.
- **Main challenge – staffing and scheduling cost:** A standalone class requires a CS specialist and a place in an already full schedule, with staffing and budget implications.

Planning Resources

Three worked examples support planning in this model, together showing how a standalone class grows across the elementary span:

- **Grade 1 examples** – Shows a full-year scope and sequence organized around eight short conceptual units at 16 hours.
- **Grade 2 examples** – Shows an early dedicated block, introducing standalone CS at 10 hours across two topics (Systems & Security; Algorithms & Design and Programming).
- **Grade 5 examples** – Shows the model at its fullest expression, spanning eight units and the full breadth of the standards at 24 hours, with deeper, project-based work.

Grade 1: Dedicated CS Class

Classroom Snapshot: A dedicated computer science specialist teaches Grade 1 CS using robots, tablets, and craft and building materials, in 30-minute weekly sessions across 32 of 36 weeks (16 hours total), organized into short conceptual units.

Grade 1: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Step-by-Step Thinking	How can we break big tasks into clear steps?	Students learn the word "algorithm" through daily routines, sequence picture cards for tasks like brushing teeth or lining up, and act out steps for classroom routines, using picture cards for differentiation. Students are assessed by explaining the steps in order, and extend their learning by inventing steps for a new class routine.	• E1-ALG-PS-01: Decompose a problem or task into individual parts to develop an algorithm.	ELA (sequencing stories), Science (procedures in experiments)
The Robot View	What happens when we follow instructions exactly?	In pairs, one student gives directions as if the other were a robot; students test sequences with arrows or movement cards and discuss how changing a step changes the outcome, using simplified commands for differentiation. Students are assessed on giving accurate instructions, and extend their learning by designing a robot path.	• E1-ALG-PS-01: Decompose a problem or task into individual parts to develop an algorithm. • E1-ALG-IM-03: Explain how a change to an algorithm leads to a different outcome.	Math (spatial directions)
Coding With Robots	How can we use commands to control a robot?	Students use classroom robots to move along a path, translate arrow algorithms into robot commands, and test sequences to reach a target, using shorter paths for differentiation. Students are assessed by whether the robot completes the task as anticipated, and extend their learning by building their own maze.	• E1-PRO-PD-04: Create code from an algorithm that includes sequence to solve a problem or express an idea. • E1-PRO-TR-07: Debug a program that includes a sequence of commands.	Math (counting steps, direction)

Grade 1: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
What Does the Code Do with Robots?	How can we explain what a program does?	Students watch a robot run a program, match its actions to a written sequence of commands, discuss what events start programs, and describe what the robot is doing step-by-step, using visual prompts for differentiation. Students are assessed through a verbal explanation of the code steps, and extend their learning by modifying commands.	• E1-PRO-RD-06: Explain the function of code that includes an event and sequence.	ELA (explaining processes)
Finding and Fixing Bugs with Robots	What should we do when instructions don't work?	The teacher shows an incorrect robot sequence; students identify the incorrect step and fix simple coding mistakes, using guided debugging for differentiation. Students are assessed by correcting errors, and extend their learning by designing bug challenges for classmates.	• E1-PRO-TR-07: Debug a program that includes a sequence of commands.	SEL (persistence)
Data About Our Class	How can we collect information to answer questions?	Students conduct a class survey (e.g., favorite fruit, pets), collect numeric and descriptive data, and create pictographs or charts, using teacher-guided data collection for differentiation. Students are assessed by interpreting class data, and extend their learning by comparing two datasets.	• E1-DAT-DC-08: Use multiple methods to collect both numeric and non-numeric data to help answer questions. • E1-DAT-DI-09: Compare a question that can be answered through a data investigation and a question that can be answered through other means. • E1-ALG-ML-02: Recognize that AI systems are technologies that use patterns in data to make decisions or generate new things.	Math (graphs and counting)

Grade 1: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Computers and Devices	What parts help computers work?	Students identify tablets, keyboards, and headphones, discuss how devices help complete tasks, and explore examples of input and output, using labeled diagrams for differentiation. Students are assessed by identifying a device's purpose, and extend their learning by comparing different devices.	• E1-SYS-HW-11: Describe the purpose of basic hardware components of a computing system, using accurate terminology. • E1-SOC-HU-15: Differentiate between activities that humans do well and activities that computing technologies do well. • E1-SOC-CE-16: Describe how computing is used by people at home, at school, and in the community.	Science (tools and technology)
People and Technology	What do people do well and what do computers do well?	Students sort tasks humans and computers perform well, discuss technologies people created to solve problems, and reflect on how computing helps in school and community, using visual charts for differentiation. Students are assessed by categorizing tasks, and extend their learning by designing a technology that solves a problem.	• E1-SOC-HU-15: Differentiate between activities that humans do well and activities that computing technologies do well. • E1-SOC-HI-14: Compare how an everyday activity changed after a specific computing technology was introduced. • E1-SYS-IM-13: Describe an individual's role in responsibly using computing systems and digital tools.	Social Studies (community helpers)

Grade 5: Dedicated CS Class

Classroom Snapshot: A dedicated CS specialist teaches Grade 5 CS using coding and machine learning platforms and physical computing devices with sensors, in 45-minute weekly sessions across 32 of 36 weeks (24 hours total).

Grade 5: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Algorithms & Computational Thinking	How do we break down complex problems and plan solutions step by step?	Students use real-life examples (e.g., planning an event) to introduce algorithms, create flowcharts with sequence, loops, and conditionals, test each other's algorithms for clarity in pairs, and use variables to track scores in simple games, with flowchart templates for differentiation. Students submit an annotated flowchart with required control structures, and extend by implementing the algorithm in code.	• E5-ALG-PS-01: Create a visual representation of an algorithm that includes variables and a combination of sequence, events, iteration, and selection to solve a problem or express an idea.	Math (order-of-operations as an algorithm) ELA (written reflection explaining algorithm design decisions)
Programming	How do we write, read, and improve code to make something work the way we intend?	Students modify a "Choose Your Own Adventure" program by adding scenes and variables, then in teams build an interactive maze game controlled by buttons, tilt, or sensors, with each student coding one part. They trace score variables, debug hardware-software issues, and document their code and controls, with starter code and pair programming for differentiation. Students are assessed on a responsive program using sequence, loops, conditionals, and a variable, with peer review, and extend by converting it to text-based code or programming a robot maze.	• E5-PRO-PD-04: Create a novel program by modifying or combining elements of existing programs. • E5-PRO-PD-05: Construct individual components of a program that are collaboratively assembled into a programming project. • E5-PRO-VD-06: Use variables to store, compare, and modify data within a program. • E5-PRO-RD-07: Create embedded or external documentation for a programming project. • E5-PRO-TR-08: Debug a program using systematic strategies.	ELA (documentation explaining program purpose and logic) Math (connecting variable expressions in code to mathematical expressions)

Grade 5: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Testing, Debugging & Iterative Design	Why is finding and fixing errors an important part of building anything?	Students debug intentionally buggy programs using a checklist, practicing one debugging strategy per session (e.g., print statements, tracing), then exchange programs, write bug reports, and reflect on how testing improved the program, with annotated code and visual tools for differentiation. Students are assessed by finding, fixing, and explaining at least two errors, and extend their learning by adding a bug to a partner's code as a challenge.	• E5-PRO-TR-08: Debug a program using systematic strategies.	Science (applying systematic testing the way scientists test solutions iteratively)
Data Collection & Analysis	How can we collect, organize, and use data to answer real-world questions?	Students design a survey, collect data using a physical computing device with sensors, import it into a spreadsheet, sort and filter it to answer questions, and create charts showing results, discussing what data can and can't tell them and the tradeoffs of apps that collect user data, with guided data collection and starter code for differentiation. Students are assessed by presenting a chart, explaining one insight, and naming one benefit and one risk of the data collected, and extend their learning by comparing two datasets for patterns.	• E5-DAT-DC-09: Use computational tools to collect and organize different types of data. • E5-DAT-DI-10: Analyze a dataset to identify the nature and possible sources of variability in the data. • E5-DAT-IM-11: Analyze the benefits and risks of a computing technology that uses collected data.	Math (interpreting line plots, graphs, and charts) Science (using data to evaluate which solution works best)

Grade 5: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Systems, Hardware & Networks	How do the parts of a computer work together, and how do computers connect to share information?	Students identify hardware parts (CPU, RAM, storage, input, output), show how hardware and software work together to complete a task, compare wired and wireless networks, learn basic cybersecurity through the CIA triad (Confidentiality, Integrity, Availability), and draw how data travels between devices, discussing how computers and the internet have changed daily life, with labeled diagrams and hands-on sorting for differentiation. Students are assessed by identifying hardware parts, explaining how data moves across a network, and sharing one way computers have changed daily life, and extend their learning by researching how the internet routes data and creating an infographic.	• E5-SYS-HW-12: Explain how hardware and software components of a computing system work together to perform input and output operations, processing, and storage. • E5-SYS-SE-13: Describe the concepts of the CIA triad and how each component is important in protecting information. • E5-SYS-NT-14: Distinguish between the components of wired and wireless networks. • E5-SYS-IM-15: Examine how computing systems impact culture and the ways people live and work.	Science (engineering thinking behind hardware component design) Social Studies (how network infrastructure shapes internet access)
AI & Machine Learning Basics	How do machines learn from data, and what happens when training data is biased?	Students train a visual machine learning model with balanced versus imbalanced data and compare results, simulate mislabeled data with cards and predict effects on output, discuss real-world sources of bias, and write a short reflection on how data quality affects fairness and how designers could improve AI for everyone, with a card-sorting activity for differentiation. Students are assessed by presenting accuracy comparisons, explaining why balanced data improves results, and sharing one way designers could make an AI tool more fair, and extend their learning by evaluating a real-world AI tool's training data quality.	• E5-ALG-ML-02: Train a machine learning model to make a classification or prediction. • E5-ALG-IM-03: Articulate how human-centered design principles are incorporated into the development of a computing technology.	Math (bar graphs comparing model accuracy before and after balancing training data) Social Studies (discussing who is responsible for ensuring AI fairness)

Grade 5: Dedicated CS Units

Unit	Essential Question	Activities	CS Standards	Integration Ideas
Data, AI, & Society	How does the data we collect and the AI tools we use affect people, both fairly and unfairly?	Students explore how recommendation systems use data, discuss benefits and risks of AI tools (e.g., smart speakers, navigation apps), design a school survey while considering ethical data use, and create a T-chart of benefits versus concerns for an AI tool, with sentence frames and graphic organizers for differentiation. Students are assessed by presenting one AI tool with at least one benefit and one concern, and extend their learning by creating a short consumer guide for responsible AI use.	• E5-DAT-IM-11: Analyze the benefits and risks of a computing technology that uses collected data. • E5-SOC-HU-18: Evaluate when it is appropriate to use or not use computing technologies to solve a problem. • E5-ALG-IM-03: Articulate how human-centered design principles are incorporated into the development of a computing technology.	Social Studies (collective responsibility in data privacy) ELA (comparing different perspectives on AI tool use)
Computing, Society, & Careers	Who creates computing technologies, who do they serve, and who could create them in the future?	Students study diverse computing innovators, explore an emerging technology and evaluate whether it helps or hurts, create a timeline showing how one invention led to the next, match their interests to CS careers, and work in a team to design an app that solves a community problem, explaining why technology is the best fit, with graphic organizers and visual designs for differentiation. Students are assessed by presenting their app idea, explaining the problem, who it helps, needed skills, and why technology is the right tool, and extend their learning by researching a CS professional and sharing what they have in common.	• E5-SOC-HI-16: Analyze how the inclusion or exclusion of diverse individuals and communities has shaped the design, development, and societal impact of computing technologies. • E5-SOC-ET-17: Examine how people decide whether or not to use emerging technologies. • E5-SOC-HU-18: Evaluate when it is appropriate to use or not use computing technologies to solve a problem. • E5-SOC-CE-19: Examine how professionals collaborate while using computing technologies to solve problems.	Social Studies (comparing computing access and innovation across eras and communities) ELA (tracing how one computing innovation led to another)

csteachers.org/pk12standards/resources