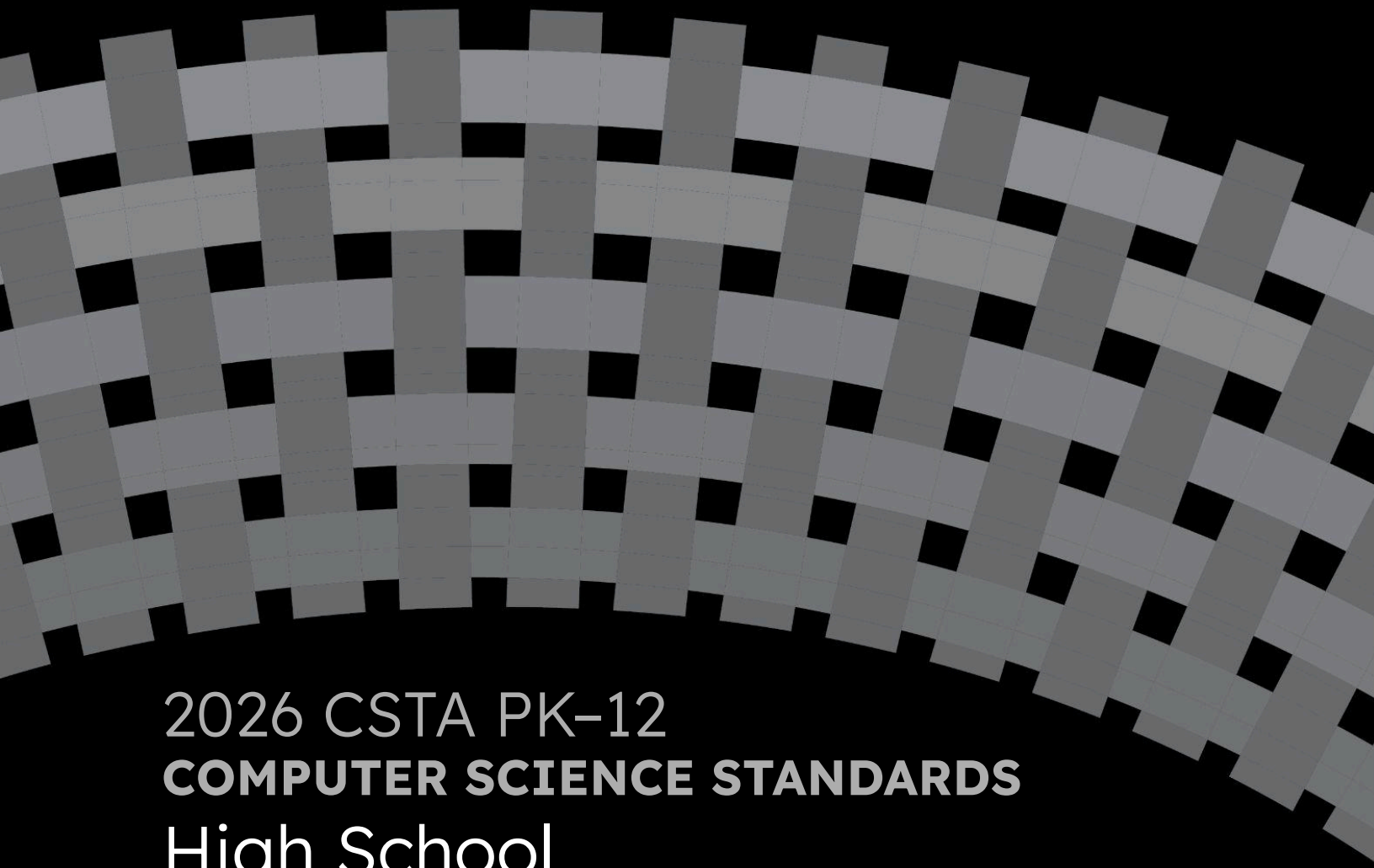
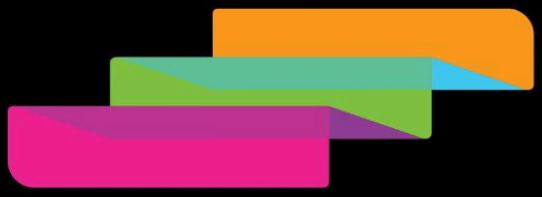


2026 CSTA PK–12

**COMPUTER SCIENCE STANDARDS**

# High School 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

This guidance is designed for instructional leadership teams making decisions about how to implement high school computer science. The standards are organized in two layers: Foundational Standards every student should learn, and Specialty Standards for deeper study in an area such as artificial intelligence, cybersecurity, game development, physical computing, software development, or data science. Because high schools vary widely in staffing, scheduling, and course catalogs, there is no single answer to how these standards should be delivered.

This document presents four models for delivering the foundational course: a full-year standalone course, a half-year course paired with core-subject integration, a full-year themed rotation, and a specialty gateway that weaves foundations into a specialty from day one. Specialty pathways are detailed later in this document. Teams should draw on the implementation examples and interdisciplinary connections included with each standard.

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

# Design Principles for High School

| Principle | Description |
|---|---|
| **Equitable Access** | Computer science should reach every student, including those who arrive without prior interest or confidence, not remain a technical elective for a self-selecting few. See the Appendix: Broadening Participation in Computer Science for how schools can put this into practice. |
| **Coherent Learning** | The standards progress from foundational to specialty, building toward deeper specialization rather than presenting a catalog of disconnected courses. Implementation should ensure every student builds the 46 Foundational Standards before or alongside specialization, so specialties deepen a shared base. Foundational exposure and specialty mastery serve different purposes: foundations are for every student, specialties for those who choose to go deeper. |
| **Flexibility** | Strong implementation can look different across schools. This document offers enrollment and sequencing options, full-year or semesterized, prerequisite or gateway, that schools can combine to fit their schedule, staffing, and course catalog while preserving the integrity of the standards. |
| **Alignment to Intent** | Implementation should reflect the full breadth of the standards, extending beyond programming alone. The Foundational Standards span algorithms, programming, data, systems, and the relationship between computing and society. A complete implementation addresses all five concepts, including the data, ethics, and societal-impact standards that are easy to omit when CS is treated as coding only. |
| **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, how courses fit the master schedule, and what support educators need. High school CS is a staffing and scheduling question as much as a curriculum one, including how foundational and specialty courses are sequenced, who is certified to teach them, and how sections are scheduled to avoid conflicts that exclude certain student groups. While there are seven specialty areas, the goal is to not offer them all; a smaller set of well-supported pathways serves students better than a broad catalog the school cannot deliver with quality. |

# Key Decisions Schools Need to Make

Successful implementation requires intentional planning about how CS functions within the high school program as a whole. Some schools already offer standalone CS or AP courses, others integrate computing into other subjects, and many are building a catalog over time. The questions below help schools assess current practice and identify the adjustments needed for alignment, coherence, and equitable access under the high school standards. Schools may find it useful to revisit these questions as implementation evolves.

1. **How do students currently experience computer science in high school?**

   Before making changes, schools should examine:
   - Which courses currently address foundational CS standards, which students take them
   - Whether access is equitable across student groups, or concentrated among a few
   - Which Foundational Standards are not yet addressed anywhere in the catalog
   - Student interest and community priorities, including the balance a community places on higher education, certifications, and direct workforce entry

2. **How will the Foundational Standards be packaged and delivered?**

   Schools should decide how all Foundational Standards reach every student, drawing on the enrollment models and delivery strategies outlined in this document, including whether courses are a semester- or-year length and whether to integrate with other content areas.

3. **How will specialties be sequenced?**

   Because specialties build on the foundations, schools should plan for:
   - Which specialty pathways the school can realistically offer given student interest
   - How specialty courses follow from or combine with the foundations

   Clear role definition supports instructional quality and consistency across grades and departments, especially in integrated models.

4. **How will access and coherence be maintained across the catalog?**

   Schools should plan for coordination so foundational and specialty courses build on each other rather than overlap or leave gaps.

The following reflection framework is designed to help leadership teams take stock of what is currently happening in high school 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 are standards not yet addressed?** | | |
| **What pathways and advanced courses do we offer?** | | |

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 trade-offs and help schools select an approach suited to their context.

## Key Decisions × Implementation Models Matrix

| Decision Area | Model 1: Full Year Foundational Course | Model 2: Semester Course & Integration | Model 3: Rotational Foundations Course | Model 4: Specialty Gateway |
|---|---|---|---|---|
| **Structure** | Standalone CS course, full year | Half-year dedicated CS course, remaining standards integrated into core subjects | Standalone CS course, full year, organized as a themed rotation | Single specialty course, full year, foundations woven throughout |
| **When Specialty Begins** | After the course ends | After the semester ends | After the course ends, with specialty previews throughout | From day one, alongside foundations |
| **Staffing** | One CS teacher for the full year | One CS teacher plus coordination with math, science, social studies, and ELA teachers | One CS teacher for the full year | One CS teacher per specialty gateway offered |
| **Best Fit** | Schools wanting a shared common base before any specialization | Schools without a full year of CS schedule space, or wanting faster specialty entry | Schools wanting a full-year course that also builds early specialty interest | Schools prioritizing engagement and recruitment through specialty appeal |

Each of these models is explored in depth in the sections that follow, including sample course sequences and planning considerations.

# Model 1: Full Year Foundational Course

Every student begins with the same dedicated foundational computer science course, covering all high school Foundational Standards before moving on to any specialty area. Once students share a base in algorithms, programming, data, systems, and the relationship between computing and society, they choose a specialty pathway such as artificial intelligence, cybersecurity, game development, physical computing, software development, or data science.

The strength of this approach is a shared common language. A student entering Advanced AI has the same grounding as one entering Cybersecurity, so specialty courses build on consistent prior knowledge rather than re-teaching fundamentals. Keeping the foundations together as one course also keeps the full set of standards visible and complete, rather than scattered across specialty contexts where a few can quietly go missing.

The foundational course runs as a standalone CS class across a full year, covering all 46 standards directly. The section that follows details a full standalone sequence that can serve as the foundational course here.

## Is This Model Right for Your School?

This model fits schools that want every CS student to share a common foundation before specializing, and that can place a dedicated foundational course ahead of specialty offerings in the sequence.

- **Main strength – a shared common base**: Every student reaches all 46 high school Foundational Standards before specializing, so specialty courses build on consistent prior knowledge and the full breadth of the foundations stays intact.
- **Main challenge – delayed specialization:** Students cannot enter a specialty until the foundational course is complete, which can postpone the high-interest work that drives engagement and requires a full year of schedule space for foundations ahead of specialties.

Schools using this model should plan how the foundational course fits the sequence and consider specialty preview modules within it to spark interest before students reach their chosen pathway.

## Semester 1: Data, Logic, and Human-Centered Design

| Topic | What It Covers | Standards |
|---|---|---|
| **Programming Mechanics** | Reading code, designing algorithms, modular programs, and collaborative workflows | • HS-ALG-PS-01: Design an algorithm using appropriate data structures to solve a problem or express an idea.<br>• HS-PRO-PD-12: Create a modular program that uses procedures, external libraries, or objects to improve reusability and readability.<br>• HS-PRO-PD-15: Collaborate on a programming project using a defined workflow that includes design documentation and clear task roles.<br>• HS-PRO-RD-17: Analyze how a segment of code works, including the role of parameters, return values, and data structures. |
| **Data Fundamentals & Collection** | Building data dictionaries, generating and cleaning data, verifying data quality | • HS-DAT-DC-21: Use a computational tool to generate simulated data that fits certain parameters for use in a simulation.<br>• HS-DAT-DC-22: Create a data dictionary that describes the name, type, and allowable values for each attribute and the logical relationships between variables in a dataset.<br>• HS-DAT-DC-23: Use a computational tool to clean and organize text-based data.<br>• HS-DAT-DC-24: Evaluate different approaches to verifying consistency and compliance with expected data types, values, and ranges. |
| **Analysis, Visualization & Ethics** | Visualizing multivariate data, evaluating results, attribution, and the ethics of large-scale data | • HS-PRO-PD-14: Apply appropriate attribution of intellectual property when developing a computing technology. HS-DAT-DI-25: Create a data visualization of a multivariate dataset to answer a question or make a classification or prediction.<br>• HS-DAT-DI-26: Evaluate a data simulation or visualization to answer a data question, inform decision-making, and identify potential limitations.<br>• HS-DAT-IM-27: Evaluate the societal, environmental, and ethical implications of large-scale data collection and processing, including within AI applications.<br>• HS-DAT-IM-28: Debate the efficacy of a policy or regulation to ensure responsible data use.<br>• HS-SOC-ET-41: Evaluate the societal and environmental impacts of an emerging technology, including those that lead to inequities in access and outcomes. |
| **Systems & Networks** | Operating systems, device capabilities, networks, and the societal impact of infrastructure | • HS-SYS-HW-29: Differentiate an operating system as a special type of software that manages both the hardware and other software components of a computing system, including handling memory and peripherals.<br>• HS-SYS-HW-30: Demonstrate the capabilities and limitations of a physical or simulated computing device to address a task or problem.<br>• HS-SYS-NT-34: Diagram a network of computing systems, including hardware and software.<br>• HS-SYS-NT-35: Analyze how the internet functions as a network of networks and how it differs from other types of networks.<br>• HS-SYS-IM-36: Evaluate the rationale behind a law or policy governing the design and use of computing systems.<br>• HS-SYS-IM-37: Investigate how computing systems and infrastructure impact society and the environment, identifying who is affected and why. |

## Semester 2: Systems, Security, and Professional Practice

| Topic | What It Covers | Standards |
|---|---|---|
| **Algorithmic Foundations & AI Selection** | Data structures, algorithm optimization and evaluation, and selecting AI approaches | • HS-ALG-PS-02: Optimize the design of an algorithm using procedural abstraction and control structures.<br>• HS-ALG-PS-03: Evaluate algorithms for efficiency, correctness, and clarity, using metrics or test cases.<br>• HS-ALG-PS-04: Describe the differences between deterministic and probabilistic algorithms.<br>• HS-ALG-ML-06: Justify the selection of a type of AI algorithm to accomplish a task.<br>• HS-ALG-PS-05: Evaluate AI-generated output to assess bias, accuracy, and potential harms.<br>• HS-PRO-VD-16: Create a program that uses appropriate data structures to store, access, and manipulate data. |
| **Advanced Development & Testing** | Development tools, evaluating AI-generated code, and refining solutions against design values | • HS-PRO-PD-13: Use documentation, libraries, application programming interfaces (APIs), and other tools in program development. HS-PRO-RD-18: Evaluate AI-generated code for accuracy, reliability, and alignment with program requirements.<br>• HS-PRO-TR-19: Evaluate a computing technology's alignment with design specifications and responsible design values, including its correctness, effectiveness, and user experience.<br>• HS-PRO-TR-20: Refine a computing technology based on user feedback, testing results, and responsible design values to improve its effectiveness and impact. |
| **Training AI & Human-Centered Design** | Training data, building ML models, human-centered design, and the values embedded in systems | • HS-ALG-ML-07: Evaluate training data by examining its source, quality, representativeness, potential biases, and privacy implications.<br>• HS-ALG-ML-08: Develop a machine learning model for a chosen task using appropriate data and tools.<br>• HS-ALG-IM-09: Design a computing technology using human-centered design principles.<br>• HS-ALG-IM-10: Evaluate the ethical implications, societal impacts, and potential biases of rule-based and data-driven algorithms. HS-ALG-IM-11: Articulate the values embedded in the design of an algorithmic system.<br>• HS-SOC-CE-45: Analyze how diverse teams of people use computational thinking and computing technologies to solve problems and express ideas.<br>• HS-SOC-CE-46: Connect computing knowledge and skills acquired to students' personal goals and career aspirations. |
| **Security, Ethics & Societal Impact** | Security measures and breaches, the history and ethics of computing, and emerging-tech impacts | • HS-SYS-SE-31: Identify different types of cybersecurity and physical security measures and the trade-offs for users, data, and devices.<br>• HS-SYS-SE-32: Classify the causes and impacts of security breaches and social engineering attacks for individuals, industries, communities, and governments.<br>• HS-SYS-SE-33: Formulate a solution to a security flaw in a given system.<br>• HS-SOC-HI-38: Analyze the historical trajectory of a specific computing technology and how its development is linked to societal and environmental factors.<br>• HS-SOC-HI-39: Propose modifications to an existing policy or piece of legislation that encourages ethical innovation and minimizes societal risks associated with technology.<br>• HS-SOC-ET-40: Evaluate the fundamental technological differences between an emerging technology and established technologies and how those differences influence computing.<br>• HS-SOC-ET-42: Design a conceptual solution to a real-world problem using an emerging technology, analyzing its potential benefits and harms.<br>• HS-SOC-HU-43: Evaluate how human choices in using, designing, deploying, and regulating computing technologies have risks, benefits, and long-term impacts.<br>• HS-SOC-HU-44: Debate perspectives on differences between human and artificial intelligence and their implications for consciousness, ethics, and human responsibility. |

# Model 2: Semester Course Plus Integrated CS Instruction

In this model, the foundational course runs for a single semester: the dedicated CS portion carries the programming, algorithms, systems, and security standards, while the data, ethics, and societal-impact standards are woven into subjects students are already taking, such as math, science, ELA, and social studies. All Foundational Standards are still met, just delivered across two settings instead of one.

Reaching a specialty sooner is the main draw. Because the dedicated course only needs a semester, students move into specialty coursework faster than a full-year model allows, without skipping any standard. The integrated placements also land where they connect naturally, data ethics alongside a statistics unit, computing history alongside social studies, so students meet the ideas in context rather than only in a CS classroom.

Coordination is what makes this work. The dedicated course and the participating subjects must run the same semester, with each subject teacher clear on which standard they own and when. The section that follows details a full split-delivery sequence, showing which standards sit in the dedicated course and which are integrated where.

## Is This Model Right for Your School?

This model fits schools that want students to reach a specialty sooner, or that don't have room in the schedule for a full year of dedicated CS, and that can coordinate with other departments to deliver the integrated standards reliably.

- **Main strength – speed to specialty:** Students complete all high school Foundational Standards in a single semester, reaching specialty coursework sooner than a full-year model allows.
- **Main challenge – cross-department coordination:** Delivery depends on other subject teachers covering their standards on schedule, which requires clear ownership and ongoing communication across departments.

## Standards Map for Standalone, Semester CS Course

| Unit | What It Covers | Standards |
|---|---|---|
| **Algorithmic Fundamentals** | Designing, optimizing, and evaluating algorithms; deterministic vs. probabilistic | • HS-ALG-PS-01: Design an algorithm using appropriate data structures to solve a problem or express an idea.<br>• HS-ALG-PS-02: Optimize the design of an algorithm using procedural abstraction and control structures.<br>• HS-ALG-PS-03: Evaluate algorithms for efficiency, correctness, and clarity, using metrics or test cases.<br>• HS-ALG-PS-04: Describe the differences between deterministic and probabilistic algorithms. |
| **Machine Learning & AI Models** | Selecting AI approaches, evaluating training data, building models, and algorithmic ethics | • HS-ALG-ML-06: Justify the selection of a type of AI algorithm to accomplish a task.<br>• HS-ALG-ML-07: Evaluate training data by examining its source, quality, representativeness, potential biases, and privacy implications.<br>• HS-ALG-ML-08: Develop a machine learning model for a chosen task using appropriate data and tools.<br>HS-ALG-IM-10: Evaluate the ethical implications, societal impacts, and potential biases of rule-based and data-driven algorithms. |
| **Program Design** | Modular programs and appropriate data structures | • HS-PRO-PD-12: Create a modular program that uses procedures, external libraries, or objects to improve reusability and readability.<br>• HS-PRO-VD-16: Create a program that uses appropriate data structures to store, access, and manipulate data. |
| **Human-Centered Design & Collaboration** | Human-centered design, collaborative workflows, and refining solutions from feedback | • HS-ALG-IM-09: Design a computing technology using human-centered design principles.<br>• HS-PRO-PD-15: Collaborate on a programming project using a defined workflow that includes design documentation and clear task roles.<br>• HS-PRO-TR-20: Refine a computing technology based on user feedback, testing results, and responsible design values to improve its effectiveness and impact. |

## Standards Map for Standalone, Semester CS Course

| Unit | What It Covers | Standards |
|---|---|---|
| **Software & Hardware Interaction** | Operating systems, device capabilities, and how networks function | • HS-SYS-HW-29: Differentiate an operating system as a special type of software that manages both the hardware and other software components of a computing system, including handling memory and peripherals.<br>• HS-SYS-HW-30: Demonstrate the capabilities and limitations of a physical or simulated computing device to address a task or problem.<br>• HS-SYS-NT-34: Diagram a network of computing systems, including hardware and software.<br>• HS-SYS-NT-35: Analyze how the internet functions as a network of networks and how it differs from other types of networks. |
| **Cybersecurity & Defense** | Security measures and trade-offs, breaches and social engineering, and fixing flaws | • HS-SYS-SE-31: Identify different types of cybersecurity and physical security measures and the trade-offs for users, data, and devices.<br>• HS-SYS-SE-32: Classify the causes and impacts of security breaches and social engineering attacks for individuals, industries, communities, and governments.<br>• HS-SYS-SE-33: Formulate a solution to a security flaw in a given system. |
| **Reading, Testing & Documentation** | Development tools, attribution, analyzing code, and evaluating AI-generated code | • HS-PRO-PD-13: Use documentation, libraries, application programming interfaces (APIs), and other tools in program development.<br>• HS-PRO-PD-14: Apply appropriate attribution of intellectual property when developing a computing technology.<br>• HS-PRO-RD-17: Analyze how a segment of code works, including the role of parameters, return values, and data structures.<br>• HS-PRO-RD-18: Evaluate AI-generated code for accuracy, reliability, and alignment with program requirements. |
| **Professionalism & Career** | Emerging-tech differences and solutions, evaluating technologies, teamwork, and career connections | • HS-PRO-TR-19: Evaluate a computing technology's alignment with design specifications and responsible design values, including its correctness, effectiveness, and user experience.<br>• HS-SOC-ET-40: Evaluate the fundamental technological differences between an emerging technology and established technologies and how those differences influence computing.<br>• HS-SOC-ET-42: Design a conceptual solution to a real-world problem using an emerging technology, analyzing its potential benefits and harms.<br>• HS-SOC-CE-45: Analyze how diverse teams of people use computational thinking and computing technologies to solve problems and express ideas.<br>• HS-SOC-CE-46: Connect computing knowledge and skills acquired to students' personal goals and career aspirations. |

# Standards Map for Integrated Instruction

| Subject Area | What It Covers | Standards |
|---|---|---|
| **Earth Science & Biology** | Generating and cleaning data, visualizing multivariate data, and infrastructure's environmental impact | • HS-DAT-DC-21: Use a computational tool to generate simulated data that fits certain parameters for use in a simulation.<br>• HS-DAT-DC-23: Use a computational tool to clean and organize text-based data.<br>• HS-DAT-DI-25: Create a data visualization of a multivariate dataset to answer a question or make a classification or prediction.<br>• HS-SYS-IM-37: Investigate how computing systems and infrastructure impact society and the environment, identifying who is affected and why. |
| **Algebra I & Statistics** | Data dictionaries, verifying data quality, and evaluating visualizations | • HS-DAT-DC-22: Create a data dictionary that describes the name, type, and allowable values for each attribute and the logical relationships between variables in a dataset.<br>• HS-DAT-DC-24: Evaluate different approaches to verifying consistency and compliance with expected data types, values, and ranges.<br>• HS-DAT-DI-26: Evaluate a data simulation or visualization to answer a data question, inform decision-making, and identify potential limitations. |
| **Government & History** | Laws and policy on computing, data ethics, history of computing, and emerging-tech impacts | • HS-DAT-IM-27: Evaluate the societal, environmental, and ethical implications of large-scale data collection and processing, including within AI applications.<br>• HS-DAT-IM-28: Debate the efficacy of a policy or regulation to ensure responsible data use.<br>• HS-SYS-IM-36: Evaluate the rationale behind a law or policy governing the design and use of computing systems.<br>• HS-SOC-HI-38: Analyze the historical trajectory of a specific computing technology and how its development is linked to societal and environmental factors.<br>• HS-SOC-HI-39: Propose modifications to an existing policy or piece of legislation that encourages ethical innovation and minimizes societal risks associated with technology.<br>• HS-SOC-ET-41: Evaluate the societal and environmental impacts of an emerging technology, including those that lead to inequities in access and outcomes.<br>• HS-SOC-HU-43: Evaluate how human choices in using, designing, deploying, and regulating computing technologies have risks, benefits, and long-term impacts. |
| **ELA** | Evaluating AI output, articulating embedded values, and debating human vs. artificial intelligence | • HS-ALG-PS-05: Evaluate AI-generated output to assess bias, accuracy, and potential harms.<br>• HS-ALG-IM-11: Articulate the values embedded in the design of an algorithmic system.<br>• HS-SOC-HU-44: Debate perspectives on differences between human and artificial intelligence and their implications for consciousness, ethics, and human responsibility. |

# Model 3: Rotational Foundations Course

The rotational model also implements all Foundational Standards in a single full-year course, but organizes them as a rotation of themed units rather than topic blocks, with each rotation explicitly bridging toward a specialty area. Where Model 1 sequences for topic coherence, this model sequences for early interest: a student gets a taste of where each rotation's skills lead before the year is over.

A sample rotation moves from algorithms and programming, to data and visualization, to systems and security, to AI and societal impact, though schools can reorder or reweight these units to fit their own calendar. Each unit closes with a deliberate bridge: programming work leads toward Software Development or Game Development, data work leads toward Data Science, systems work leads toward Cybersecurity, and AI and history standards lead toward AI or Physical Computing. A capstone activity, such as using an emerging-technology standard to have students design a conceptual "smart" device, can serve as the bridge into whichever specialty a student found most compelling.

The AI and societal-impact unit tends to carry more standards than the others, since the history and societal-impact standards work best as a capstone reflection on everything the course has covered, rather than being distributed earlier.

## Is This Model Right for Your School?

This model fits schools that want a single full-year foundational course, like Model 1, but want that course to actively build interest in specific specialties as it goes, rather than waiting until the course ends to introduce them.

- **Main strength – early specialty interest:** Students see where their skills are heading throughout the year, which can sharpen engagement and give students a head start deciding which specialty to pursue.
- **Main challenge – uneven pacing:** The rotation's units don't carry equal weight; the AI and societal-impact unit in particular is dense, so pacing and depth have to be managed deliberately rather than assuming each unit is interchangeable.

Schools using this model should treat the specialty bridges as previews, not full specialty instruction, and should plan the heavier final unit into the pacing calendar from the start, adjusting the rotation's order and weighting to fit their own schedule.

## Sample Rotational Unit Design

| Unit | What It Covers | Standards |
|---|---|---|
| **Algorithms & Programming** | Building the mental models for how code works, moving from a logical plan to modular, readable programs.<br><br>Bridges toward **Software Development** or **Game Development.** | • HS-ALG-PS-01: Design an algorithm using appropriate data structures to solve a problem or express an idea.<br>• HS-ALG-PS-02: Optimize the design of an algorithm using procedural abstraction and control structures.<br>• HS-ALG-PS-03: Evaluate algorithms for efficiency, correctness, and clarity, using metrics or test cases.<br>• HS-ALG-PS-04: Describe the differences between deterministic and probabilistic algorithms.<br>• HS-PRO-PD-12: Create a modular program that uses procedures, external libraries, or objects to improve reusability and readability.<br>• HS-PRO-PD-13: Use documentation, libraries, application programming interfaces (APIs), and other tools in program development.<br>• HS-PRO-PD-14: Apply appropriate attribution of intellectual property when developing a computing technology.<br>• HS-PRO-PD-15: Collaborate on a programming project using a defined workflow that includes design documentation and clear task roles.<br>• HS-PRO-RD-17: Analyze how a segment of code works, including the role of parameters, return values, and data structures. |
| **Data & Visualization** | Transitioning from code to content, learning how to clean, organize, and visualize data to tell a story or make a prediction.<br><br>Bridges toward **Data Science**. | • HS-DAT-DC-21: Use a computational tool to generate simulated data that fits certain parameters for use in a simulation.<br>• HS-DAT-DC-22: Create a data dictionary that describes the name, type, and allowable values for each attribute and the logical relationships between variables in a dataset.<br>• HS-DAT-DC-23: Use a computational tool to clean and organize text-based data.<br>• HS-DAT-DC-24: Evaluate different approaches to verifying consistency and compliance with expected data types, values, and ranges.<br>• HS-DAT-DI-25: Create a data visualization of a multivariate dataset to answer a question or make a classification or prediction.<br>• HS-DAT-DI-26: Evaluate a data simulation or visualization to answer a data question, inform decision-making, and identify potential limitations.<br>• HS-DAT-IM-27: Evaluate the societal, environmental, and ethical implications of large-scale data collection and processing, including within AI applications.<br>• HS-DAT-IM-28: Debate the efficacy of a policy or regulation to ensure responsible data use. |

## Sample Rotational Unit Design

| Unit | What It Covers | Standards |
|---|---|---|
| **Systems & Security** | Understanding the physical and virtual infrastructure, exploring how computers communicate and how those communications are kept private.<br><br>Bridges toward **Cybersecurity**. | • HS-SYS-HW-29: Differentiate an operating system as a special type of software that manages both the hardware and other software components of a computing system, including handling memory and peripherals.<br>• HS-SYS-HW-30: Demonstrate the capabilities and limitations of a physical or simulated computing device to address a task or problem.<br>• HS-SYS-SE-31: Identify different types of cybersecurity and physical security measures and the trade-offs for users, data, and devices.<br>• HS-SYS-SE-32: Classify the causes and impacts of security breaches and social engineering attacks for individuals, industries, communities, and governments.<br>• HS-SYS-SE-33: Formulate a solution to a security flaw in a given system.<br>• HS-SYS-NT-34: Diagram a network of computing systems, including hardware and software.<br>• HS-SYS-NT-35: Analyze how the internet functions as a network of networks and how it differs from other types of networks.<br>• HS-SYS-IM-36: Evaluate the rationale behind a law or policy governing the design and use of computing systems.<br>• HS-SYS-IM-37: Investigate how computing systems and infrastructure impact society and the environment, identifying who is affected and why. |
| **AI & Societal Impact** | The modern frontier: evaluating AI-generated work, analyzing bias, and connecting learning to careers and history.<br><br>Bridges toward **AI** or **Physical Computing**. | • HS-ALG-PS-05: Evaluate AI-generated output to assess bias, accuracy, and potential harms.<br>• HS-ALG-ML-06: Justify the selection of a type of AI algorithm to accomplish a task.<br>• HS-ALG-ML-07: Evaluate training data by examining its source, quality, representativeness, potential biases, and privacy implications.<br>• HS-ALG-ML-08: Develop a machine learning model for a chosen task using appropriate data and tools.<br>• HS-ALG-IM-09: Design a computing technology using human-centered design principles.<br>• HS-ALG-IM-10: Evaluate the ethical implications, societal impacts, and potential biases of rule-based and data-driven algorithms.<br>• HS-ALG-IM-11: Articulate the values embedded in the design of an algorithmic system.<br>• HS-PRO-VD-16: Create a program that uses appropriate data structures to store, access, and manipulate data.<br>• HS-PRO-RD-18: Evaluate AI-generated code for accuracy, reliability, and alignment with program requirements.<br>• HS-PRO-TR-19: Evaluate a computing technology's alignment with design specifications and responsible design values, including its correctness, effectiveness, and user experience. |

## Sample Rotational Unit Design

| Unit | What It Covers | Standards |
|---|---|---|
| **AI & Societal Impact** (continued) | | • HS-PRO-TR-20: Refine a computing technology based on user feedback, testing results, and responsible design values to improve its effectiveness and impact.<br>• HS-SOC-HI-38: Analyze the historical trajectory of a specific computing technology and how its development is linked to societal and environmental factors.<br>• HS-SOC-HI-39: Propose modifications to an existing policy or piece of legislation that encourages ethical innovation and minimizes societal risks associated with technology.<br>• HS-SOC-ET-40: Evaluate the fundamental technological differences between an emerging technology and established technologies and how those differences influence computing.<br>• HS-SOC-ET-41: Evaluate the societal and environmental impacts of an emerging technology, including those that lead to inequities in access and outcomes.<br>• HS-SOC-ET-42: Design a conceptual solution to a real-world problem using an emerging technology, analyzing its potential benefits and harms.<br>• HS-SOC-HU-43: Evaluate how human choices in using, designing, deploying, and regulating computing technologies have risks, benefits, and long-term impacts.<br>• HS-SOC-HU-44: Debate perspectives on differences between human and artificial intelligence and their implications for consciousness, ethics, and human responsibility.<br>• HS-SOC-CE-45: Analyze how diverse teams of people use computational thinking and computing technologies to solve problems and express ideas.<br>• HS-SOC-CE-46: Connect computing knowledge and skills acquired to students' personal goals and career aspirations. |

# Model 4: Specialty Gateway

Rather than starting with a generic foundations course, students enter computer science through a single specialty. A course like Introduction to AI teaches the Foundational Standards through that specialty's lens while building specialty knowledge in the same sequence: students learn algorithm design by training a simple model, then refine that model's accuracy as a specialty skill built on the same foundation. Foundational and Specialty Standards run together across one full-year course, not as a foundations unit followed by a specialty add-on.

The appeal is immediate engagement. Students who might never enroll in a course called Foundations of CS may sign up for AI, Game Development, or Cybersecurity, building both the shared foundation and specialty-level skill once inside. The tradeoff is coverage: because foundations are woven into one specialty's context rather than taught neutrally, a school offering several gateways must confirm each one still delivers the full foundational set alongside its specialty content. The section that follows shows one full year of Intro to AI, with foundational and specialty content integrated into a single sequence, as an illustration of what a gateway course needs to deliver.

## Is This Model Right for Your School?

This model fits schools prioritizing engagement and recruitment, especially those trying to broaden participation by meeting students through interests they already hold.

- **Main strength – immediate interest:** Students enter through a specialty they find compelling and build both foundational and specialty skill in a context that already feels relevant, which supports recruitment and retention.
- **Main challenge – ensuring full coverage:** Since foundations are woven into a specialty course rather than taught separately, each gateway must be designed to cover all high school Foundational Standards, not only those that fit its theme, which requires deliberate mapping.

## Illustrative Example: Intro to AI Gateway Course

The table below shows one way a full-year Intro to AI course integrates all high school Foundational Standards with the AI Specialty I Standards, so students who enrolled for AI build the complete foundation and specialty-level skill in a single course. The same approach works for an Intro to Cybersecurity, Intro to Game Development, or any other specialty area; each must integrate Specialty Standards content with the set of full Foundational Standards.

## Intro to AI Gateway Course (Full Year)

| Unit | Description | Standards Introduced |
|---|---|---|
| **1. AI and A World Powered by Computing** | Establishes a shared understanding of what AI is, how it differs from other computing approaches, and why its development matters. Students begin examining the human choices, values, and consequences that they will revisit throughout the course. | • HS-ALG-PS-04: Describe the differences between deterministic and probabilistic algorithms.<br>• HS-ALG-PS-05: Evaluate AI-generated output to assess bias, accuracy, and potential harms.<br>• HS-SOC-HI-38: Analyze the historical trajectory of a specific computing technology and how its development is linked to societal and environmental factors.<br>• HS-SOC-ET-40: Evaluate the fundamental technological differences between an emerging technology and established technologies and how those differences influence computing.<br>• HS-SOC-ET-41: Evaluate the societal and environmental impacts of an emerging technology, including those that lead to inequities in access and outcomes.<br>• HS-SOC-HU-44: Debate perspectives on differences between human and artificial intelligence and their implications for consciousness, ethics, and human responsibility.<br>• *S1-AIN-DD-01: Analyze AI systems to differentiate the types of problems they address.*<br>• *S1-AIN-DD-05: Evaluate whether an AI or non-AI computational solution is appropriate for a real-world problem.* |
| **2. Algorithms and Program Development** | Builds the algorithmic thinking and programming skills needed to understand and create computational systems. Students develop a foundation in program structure, collaboration, and evaluation before applying these skills within data-driven and AI applications. | • HS-ALG-PS-01: Design an algorithm using appropriate data structures to solve a problem or express an idea.<br>• HS-ALG-PS-02: Optimize the design of an algorithm using procedural abstraction and control structures.<br>• HS-ALG-PS-03: Evaluate algorithms for efficiency, correctness, and clarity, using metrics or test cases.<br>• HS-PRO-PD-12: Create a modular program that uses procedures, external libraries, or objects to improve reusability and readability.<br>• HS-PRO-PD-13: Use documentation, libraries, application programming interfaces (APIs), and other tools in program development.<br>• HS-PRO-PD-14: Apply appropriate attribution of intellectual property when developing a computing technology.<br>• HS-PRO-PD-15: Collaborate on a programming project using a defined workflow that includes design documentation and clear task roles.<br>• HS-PRO-VD-16: Create a program that uses appropriate data structures to store, access, and manipulate data.<br>• HS-PRO-RD-17: Analyze how a segment of code works, including the role of parameters, return values, and data structures. |

## Intro to AI Gateway Course (Full Year)

| Unit | Description | Standards Introduced |
|---|---|---|
| **3. Data Foundations for AI** | Develops students' understanding of how data is represented, prepared, and interpreted. This work establishes the data literacy needed to recognize quality and representation issues before students begin training and evaluating machine learning models. | • HS-DAT-DC-21: Use a computational tool to generate simulated data that fits certain parameters for use in a simulation.<br>• HS-DAT-DC-22: Create a data dictionary that describes the name, type, and allowable values for each attribute and the logical relationships between variables in a dataset.<br>• HS-DAT-DC-23: Use a computational tool to clean and organize text-based data.<br>• HS-DAT-DC-24: Evaluate different approaches to verifying consistency and compliance with expected data types, values, and ranges.<br>• HS-DAT-DI-25: Create a data visualization of a multivariate dataset to answer a question or make a classification or prediction.<br>• HS-DAT-DI-26: Evaluate a data simulation or visualization to answer a data question, inform decision-making, and identify potential limitations.<br>▪ *S1-AIN-DD-04: Compare data representations and how representation choice constrains applicable algorithms.*<br>▪ *S1-AIN-DS-07: Apply data acquisition, cleaning, and transformation techniques to prepare data for AI analysis.* |
| **4. Systems, Networks, and Security** | Examines the hardware, software, networks, and security structures that allow computing and AI systems to operate. Students consider both technical infrastructure and safeguards before designing applications that collect data or affect users. | • HS-SYS-HW-29: Differentiate an operating system as a special type of software that manages both the hardware and other software components of a computing system, including handling memory and peripherals.<br>• HS-SYS-HW-30: Demonstrate the capabilities and limitations of a physical or simulated computing device to address a task or problem.<br>• HS-SYS-SE-31: Identify different types of cybersecurity and physical security measures and the trade-offs for users, data, and devices.<br>• HS-SYS-SE-32: Classify the causes and impacts of security breaches and social engineering attacks for individuals, industries, communities, and governments.<br>• HS-SYS-SE-33: Formulate a solution to a security flaw in a given system.<br>• HS-SYS-NT-34: Diagram a network of computing systems, including hardware and software.<br>• HS-SYS-NT-35: Analyze how the internet functions as a network of networks and how it differs from other types of networks. |

## Intro to AI Gateway Course (Full Year)

| Unit | Description | Standards Introduced |
|---|---|---|
| **4. Systems, Networks, and Security** *(continued)* | | • HS-SYS-IM-36: Evaluate the rationale behind a law or policy governing the design and use of computing systems.<br>• HS-SYS-IM-37: Investigate how computing systems and infrastructure impact society and the environment, identifying who is affected and why.<br>▪ *S1-AIN-HR-08: Plan safeguards for AI systems that protect human well-being and privacy while ensuring meaningful human involvement in decision-making.* |
| **5. Machine Learning: From Data to Models** | Connects earlier learning about algorithms and data to the development of machine learning models. Students move from selecting an appropriate approach to training and evaluating models, preparing them to use AI purposefully within applications. | • HS-ALG-ML-06: Justify the selection of a type of AI algorithm to accomplish a task.<br>• HS-ALG-ML-07: Evaluate training data by examining its source, quality, representativeness, potential biases, and privacy implications.<br>• HS-ALG-ML-08: Develop a machine learning model for a chosen task using appropriate data and tools.<br>• S1-AIN-DD-02: Modify AI system training data to improve fairness and accuracy in outputs.<br>• S1-AIN-DD-03: Create an application using a prebuilt supervised learning model to make a classification or prediction.<br>▪ *S1-AIN-HR-09: Analyze the potential biases and limitations of AI systems.* |
| **6. Human-Centered AI Applications** | Brings together programming, machine learning, and human-centered design as students develop and refine an AI-enabled application. The unit emphasizes that technical performance alone is insufficient; applications must also respond to users' needs, experiences, and values. | • HS-ALG-IM-09: Design a computing technology using human-centered design principles.<br>• HS-PRO-RD-18: Evaluate AI-generated code for accuracy, reliability, and alignment with program requirements.<br>• HS-PRO-TR-19: Evaluate a computing technology's alignment with design specifications and responsible design values, including its correctness, effectiveness, and user experience.<br>• HS-PRO-TR-20: Refine a computing technology based on user feedback, testing results, and responsible design values to improve its effectiveness and impact.<br>▪ *S1-AIN-PP-12: Analyze how AI tools shape user experiences for people with diverse backgrounds and characteristics.* |

## Intro to AI Gateway Course (Full Year)

| Unit | Description | Standards Introduced |
|---|---|---|
| **7. Responsible AI, Data, and Technology Policy** | Broadens the focus from individual applications to the systems, institutions, and policies surrounding AI. Students use their technical understanding to examine impacts, responsibility, data practices, and governance before making design decisions in the capstone. | • HS-ALG-IM-10: Evaluate the ethical implications, societal impacts, and potential biases of rule-based and data-driven algorithms.<br>• HS-ALG-IM-11: Articulate the values embedded in the design of an algorithmic system.<br>• HS-DAT-IM-27: Evaluate the societal, environmental, and ethical implications of large-scale data collection and processing, including within AI applications.<br>• HS-DAT-IM-28: Debate the efficacy of a policy or regulation to ensure responsible data use.<br>• HS-SOC-HI-39: Propose modifications to an existing policy or piece of legislation that encourages ethical innovation and minimizes societal risks associated with technology.<br>• HS-SOC-HU-43: Evaluate how human choices in using, designing, deploying, and regulating computing technologies have risks, benefits, and long-term impacts.<br>▪ *S1-AIN-HR-10: Analyze the environmental impacts of widespread AI adoption.*<br>▪ *S1-AIN-PP-13: Assess how unauthorized data collection has influenced the practice of training AI models.*<br>▪ *S1-AIN-PP-14: Evaluate how ethical implications of AI have changed over time.* |
| **8. Capstone: Designing a Responsible AI-Enabled Solution** | Synthesizes learning from across the course through a sustained design challenge. Students apply technical, ethical, and human-centered reasoning to propose or create a responsible solution and communicate the choices and trade-offs behind it. | • HS-SOC-ET-42: Design a conceptual solution to a real-world problem using an emerging technology, analyzing its potential benefits and harms.<br>• HS-SOC-CE-45: Analyze how diverse teams of people use computational thinking and computing technologies to solve problems and express ideas.<br>• HS-SOC-CE-46: Connect computing knowledge and skills acquired to students' personal goals and career aspirations. |

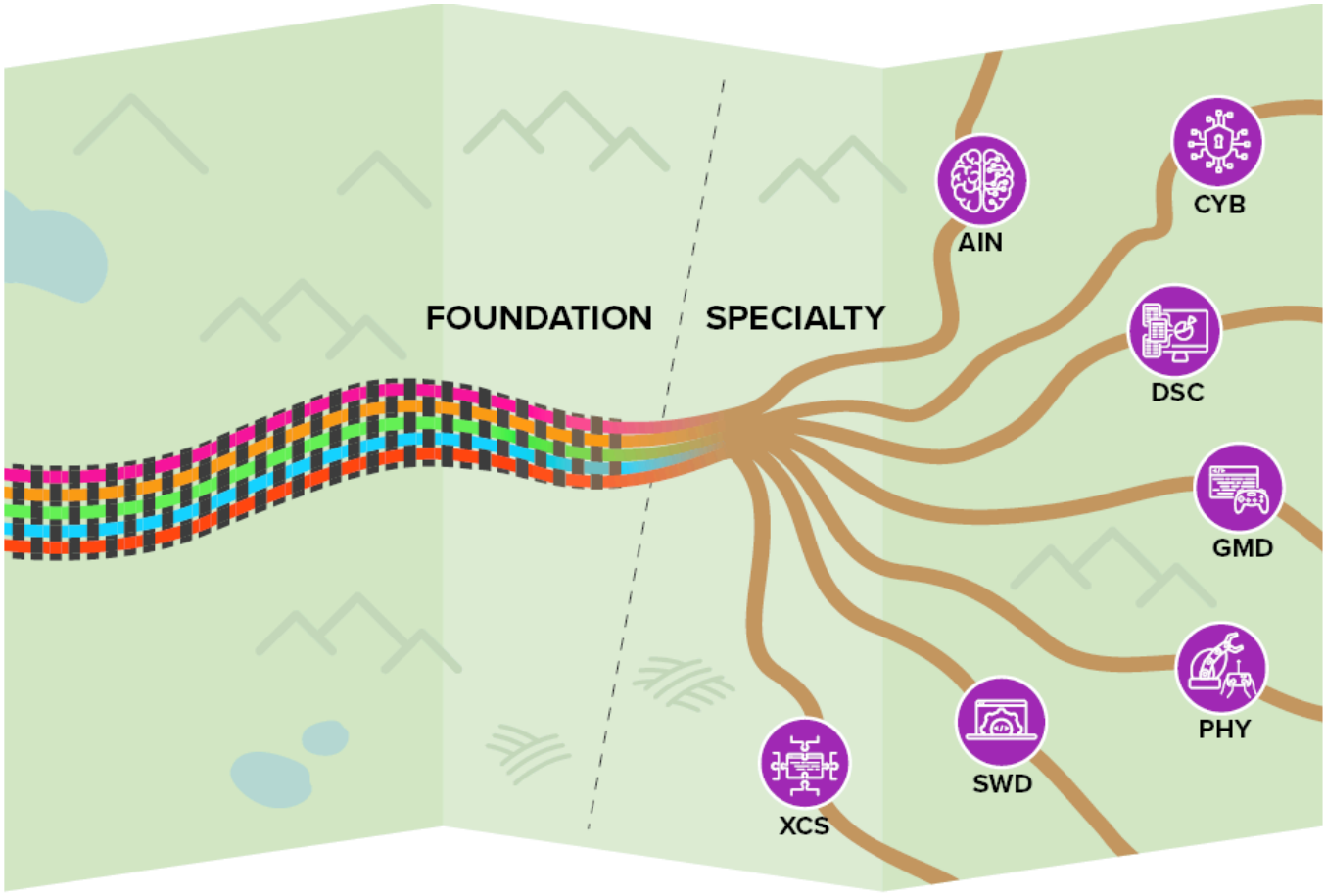
Note: Standards are listed with the unit in which they are introduced or most directly assessed, but learning should build naturally across the course, with concepts and skills reinforced in subsequent units. Practices such as collaboration, documentation, attribution, testing, responsible design, and consideration of impacts should be integrated throughout. This course design includes 12 of 14 AI Specialty I standards. Two additional AI Specialty I standards may be included as optional extensions: *S1-AIN-DS-06: Examine how data flows through a neural network structure.* in Unit 5, and *S1-AIN-PP-11: Integrate a prebuilt AI agent into an application.* in Unit 6.

# Developing Specialty Pathways

## Specialty Standards

Once students complete a strong foundation in computer science, they are able to pursue **advanced, domain-specific learning**. The 2026 CSTA PK–12 Standards define a set of High School Specialty Standards. These standards are organized into two tiers (Specialty I and Specialty II) across six high school specialty areas identified through the *Reimagining CS Pathways* project:

- Artificial Intelligence (AIN)
- Cybersecurity (CYB)
- Data Science (DSC)
- Game Development (GMD)
- Physical Computing (PHY)
- Software Development (SWD)




**Specialty I** standards introduce the knowledge and skills essential within a chosen area, serving as a student's first focused learning experience in that domain. **Specialty II** standards describe more advanced study, preparing students for postsecondary coursework, industry certifications, or continued specializations. Additionally, **X+CS** Standards support interdisciplinary learning by integrating foundational high school CS content into other subject areas, such as journalism, biology, and the arts.

## Specialty Areas at a Glance

| Specialty Area | Specialty I Coverage | Specialty II Coverage |
|---|---|---|
| **Artificial Intelligence** | Differentiate AI problem types, prepare data for models, build applications with prebuilt models, and plan safeguards for human well-being and privacy. | Design solutions using unsupervised learning, develop and evaluate models, and address the environmental and ethical implications of AI systems. |
| **Cybersecurity** | Analyze network services, protocols, and vulnerabilities; evaluate threats using layered defense; and explain the role of cybersecurity policies. | Design secure networks, build access controls and incident response plans, and evaluate how regulations shape organizational security. |
| **Data Science** | Apply exploratory analysis to structured data, interpret models and results, create visualizations, and apply ethical principles to data use. | Analyze unstructured and high-dimensional data, build and evaluate models, and address legal and social considerations at scale. |
| **Game Development** | Analyze game components, storyboard and build interactive experiences, design usable interfaces, and collaborate on a game project. | Program animation and NPC behavior, evaluate immersion and usability, and design scalable, maintainable game architectures. |
| **Physical Computing** | Construct circuits, program inputs and outputs, and build connected physical computing devices. | Develop electromechanical systems with design tools, extend device connectivity, and refine systems through testing. |
| **Software Development** | Implement linear data structures, design user experiences, and test and refine programs. | Apply associative and hierarchical data structures, advance user-experience design, and evaluate software against defined criteria. |
| **X+CS** | Integrate foundational high school CS content into another subject, such as journalism, biology, or the arts, applying computing within a field students are already studying. | |

The Specialty Standards build from the content of the Foundational Standards. **This spreadsheet** shows specific progressions from the High School Foundational Standards to each set of the High School Specialty Standards.

## How do Specialty Standards differ from Foundational Standards?

The Foundational Standards prepare every student for a world powered by computing, whereas the Specialty Standards extend this foundation by supporting deeper learning in specific areas of computer science for students who choose to continue their studies, often through courses and pathways. The core differences between Foundational and Specialty Standards lie in the target audience, goal, and scope and are summarized in the following table.

| | Foundational Standards | Specialty Standards |
|---|---|---|
| **Target Audience** | **All** PK–12 students. | High school students who pursue continued CS learning **beyond** the foundational PK–12 standards. |
| **Primary Goal** | Develop CS knowledge, skills, and dispositions for **informed participation in society**, critical content consumption, responsible creation, and general problem-solving. | Develop **deeper, domain-specific** knowledge, skills, and dispositions that align with student interests and related pathways to college and/or career. |
| **Scope** | **Broad content** across five concepts: Algorithms & Design, Programming, Data & Analysis, Systems & Security, and Computing & Society. | **Focused, advanced** content within a specific computing domain (e.g., software development, artificial intelligence, data science). |

## The Relationship Between Specialty Standards and CTE

Specialty Standards and Career and Technical Education (CTE) frameworks serve complementary purposes. Specialty Standards define advanced CS learning, and CTE emphasizes workforce preparation and industry credentials. Specialty Standards can:

- Align with existing CTE pathways,
- Supplement CTE courses and pathways, and
- Inform the refinement of CTE standards and curricula to increase the depth and breadth of CS content.

Specialty Standards are not exclusively intended to complement CTE experiences. They may also inform the development of academic courses and pathways, integrated studies, and informal or nontraditional learning experiences (e.g., out-of-school time).

# How to Implement Specialty Standards

Specialty Standards can be implemented in multiple ways depending on local context, resources, and student interest. Like the Foundational Standards, they can be implemented through standalone experiences, integrated with other disciplines, or a combination.

1. **Develop Course Pathways:** Package the standards into discrete, sequential course pathways. These could focus on one specialty area or combine multiple areas.

| Strategy | Course 1 | Course 2 |
|---|---|---|
| **Exploratory Specialty Pathway** | • **Course Title:** Programming the Future<br>• **Standards:** Subsets of Specialty I standards across Software Development, Artificial Intelligence, Cybersecurity, and Data Science<br>• **Sample rationale for offering:** Many students enjoyed their foundational CS experience and want to continue their learning. They want to maintain a broad view of computer science before they dive more deeply into a singular subdomain. | • **Course Title:** Information and Network Security<br>• **Standards:** Cybersecurity Specialty II (and any Cybersecurity Specialty I standards that were not covered in Programming the Future)<br>• **Sample rationale for offering:** After offering Programming the Future, student interest seemed to favor cybersecurity, so an additional course opportunity was created in this area. |
| **Focused Specialty Pathway** | • **Course Title:** AI Fundamentals<br>• **Standards:** AI Specialty I<br>• **Sample rationale for offering:** The local community, including parents and employers, prioritize learning opportunities to build specialized knowledge in AI. | • **Course Title:** Developing AI Applications<br>• **Standards:** AI Specialty II<br>• **Sample rationale for offering:** Students developed their skills and confirmed their interest in AI through AI Fundamentals and wanted to continue to learn even more about how AI works and how to develop AI applications. |

2. **Augment a Foundational Course to Emphasize One or More Specialty Areas:** Incorporate content from a specialty area into foundational CS learning experiences to increase early exposure to specialty areas and/or create thematic foundational experiences. This might include using AI Specialty Standards to develop an AI unit that is incorporated into a foundational CS course or using Cybersecurity Specialty Standards to dive deeper into Systems & Security concepts within a foundational experience. For specific examples, see Model 3: Rotational Foundations Course and Model 4: Specialty Gateway.

3. **Guide Integrated Courses/Experiences:** Use the Specialty Standards to structure interdisciplinary courses/experiences that blend CS with other subjects (e.g., Computational Biology or Computational Journalism), ensuring the CS depth goes beyond the foundational level. This approach requires co-requisite knowledge in the non-CS discipline (X). X+CS specialty standards could be used across any discipline, however other specialty area standards can inform how CS concepts might be integrated into other disciplines as well (e.g., leverage Data Science Specialty Standards to integrate CS into math coursework). The table below shows a course progression that infuses CS and journalism:

| Foundational Courses | X+CS Course |
|---|---|
| **Computer Science Foundations**<br><br>Computer Science Foundations supports all high school students, regardless of postsecondary goals, in developing the knowledge, skills, and dispositions necessary to navigate and understand the technology-driven world in which they live. Course content is organized into five concepts: Algorithms & Design, Programming, Data & Analysis, Systems & Security, and Computing & Society.<br><br>**Introduction to Journalism**<br><br>Introduction to Journalism includes the fundamentals of gathering, writing, and reporting news stories across various media platforms. Students learn essential skills including conducting interviews, fact-checking, understanding media ethics and law, and recognizing different story formats (e.g., news articles, features, opinion pieces). The course typically emphasizes critical thinking, effective communication, and the role of journalism in a democratic society. | **Computational Journalism**<br><br>Designed for students who have completed a foundational computing course as well as a foundational journalism course, this class exposes students to computational techniques and issues related to journalism, including:<br><br>• Ethical issues, including data privacy and security<br>• Language processing and text analysis<br>• Reporting on technology and the technology industry<br>• Data journalism, including data visualization<br>• AI and its impact on the field of journalism |

4. **Offer content as part of (or in tandem with) one or more existing programs:** It may be advantageous for schools to build on existing infrastructure in support of CS programs. This can reduce barriers to entry and infuse some level of familiarity into CS implementation. Existing programs may include:

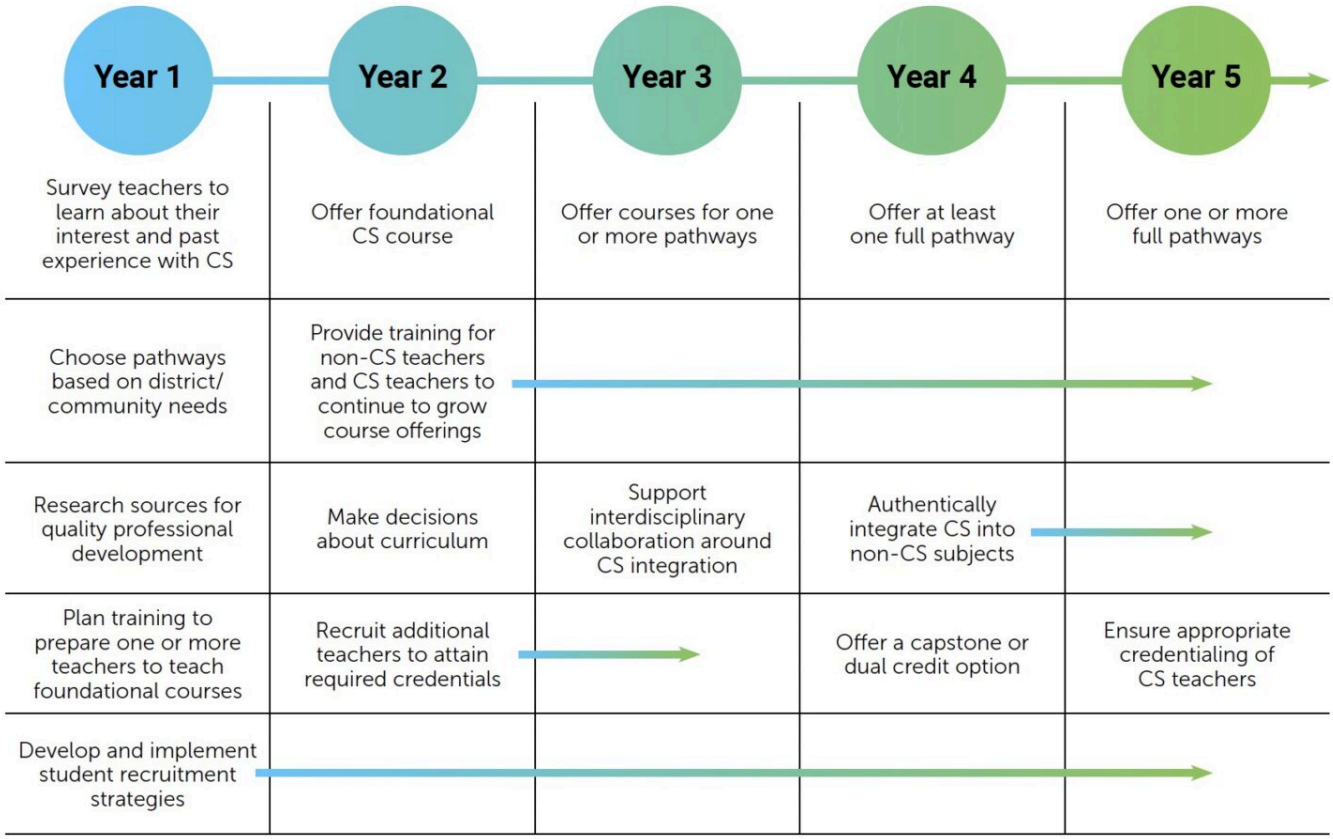
   - Advanced Placement (AP)
   - Career and Technical Education (CTE)
   - International Baccalaureate (IB)
   - Virtual or online course offerings
   - Partnership with an institution of higher education for early college credit (i.e., dual enrollment, dual credit)
   - Relevant out-of-school programming

# Implementation Planning

Before implementing a new CS pathway, it is important to define current CS offerings at a school or district and then to identify areas for development based on relevance to the community, available resources, and desired outcomes for students. Ramping up a robust CS program takes time, and it may require establishing a multiyear plan that involves assessing teacher interest, evaluating possible professional development opportunities, training teachers (including non-CS teachers), and recruiting additional teachers.

Due to resource constraints, very few schools will be able to offer the full range of Specialty Standards. However, combining several ways to offer content can extend beyond the classroom; for example, small schools may offer out-of-school activities like robotics club or e-textiles. They may also establish partnerships within the community to offer summer enrichment camps and programs, apprenticeship programs, and other resources to support students in their learning. The process of determining implementation pathways is not simple, as a wide variety of factors—from teacher capacity to student interest to local needs—must be considered.

The following figure shows an example of how a school may design and implement a set of relevant CS pathways over the course of five years.



# Equity Considerations for CS Pathways

**The importance of flexibility.** Flexibility better supports students who, for example, move into a school district in the middle of high school to participate in the pathway. Flexibility is also useful for students who choose to change pathways, have differing prior experience, extend learning outside of school, or complete self-guided learning. Further, pathways can be created to accommodate a range of prior experiences and a variety of postsecondary plans, including not just higher education but also industry certifications, direct entry into the workforce, and military service.

**Resource limitations.** While highly resourced schools may be able to implement a wide variety of CS pathways and options, students in other types of schools may have fewer opportunities to exercise choice in what CS content to study. Education leaders can make every effort to ensure that resources are available to implement CS pathways that meet student interests, their community needs, and available resources. Innovative solutions may need to be implemented to overcome barriers specific to rural and urban contexts (e.g., teacher sharing programs, transportation for after-school programs).

**Alignment with other programs and entities.** Course offerings are often connected to teacher certification/credentialing requirements, which may limit a school's ability to offer specific courses. For example, high school CS courses are often classified as either CTE or traditional academic courses. In some states, dual coding is permitted, and in others, it is not. Offering CTE courses may qualify schools for Perkins V funding to support software, hardware/equipment, curricular materials, teacher professional development, and hiring of new teachers and administrators for up to three years. Opportunities for postsecondary credit (e.g., dual enrollment) and placement in advanced coursework (e.g., after passing AP exams) may be limited by students' ability to pay for college credits, exams, and certifications. Finally, communities place differing priorities on higher education versus certifications, which will impact schools' selection of programs.

**Gate-opening vs. gatekeeping.** CS teachers identify a lack of support, interest, or knowledge by administrators and counselors as one of the greatest challenges to teaching and promoting equity in CS education. Those who schedule courses have a tremendous impact on student participation. For example, misunderstandings lead counselors and administrators to not suggest or recommend CS to students with disabilities. It is critical that educators view CS as foundational for all students and support them in pursuing relevant pathways of study.

**Course names and descriptions matter.** Choosing names for CS courses has been identified as a promising practice for encouraging students from traditionally underrepresented groups to pursue computing. At the same time, there is often a tension between choosing names that are familiar to most students (e.g., "Game Design") and choosing names that may be more appealing to students less likely to fit stereotypes about who CS is for (e.g., "Interactive Media"). Regardless of the name chosen, it is important to ensure that courses appeal widely and that all students, teachers, administrators, and counselors understand what the courses offer.

# Example Pathways

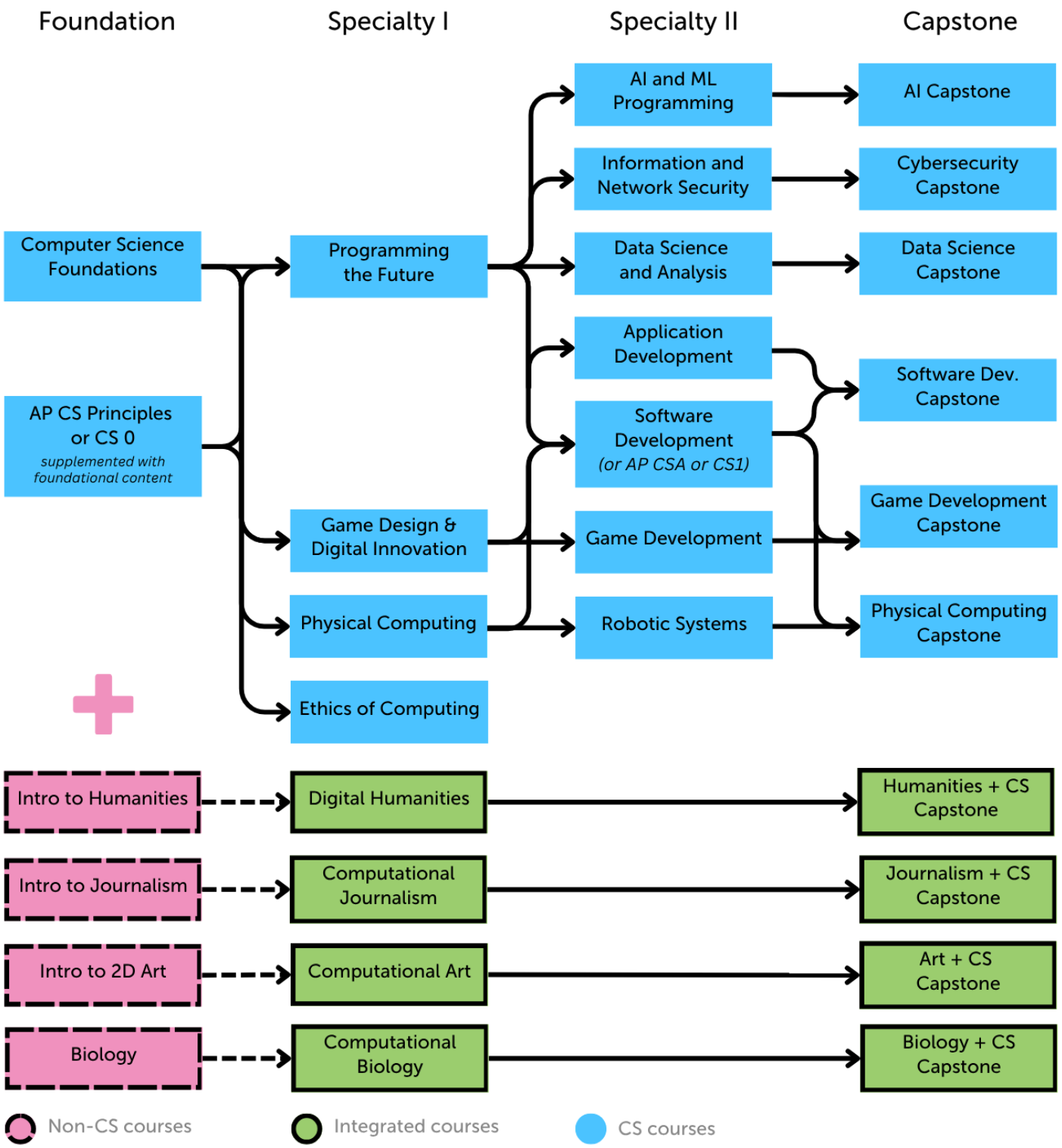
The figure on the next page illustrates a sample set of course implementation pathways. Actual pathways can and should differ widely based on local needs and resources. However, this sample is meant to suggest what a relatively full implementation might look like for a large school positioned to deliver a comprehensive set of CS pathways. It is not expected that schools would necessarily have the capability of offering all (or even multiple) of the pathways represented.

In this model, students experience the foundational CS content through taking a course called *Computer Science Foundations* (or an equivalent CS experience may be substituted). Specialty I content from different specialty areas is packaged both discretely (e.g., *Physical Computing* and *Game Design & Digital Innovation*) or in a combined fashion (e.g., the *Programming the Future* course includes Specialty I standards from AI, Cybersecurity, Data Science, and Software Development). Since students may take courses in different grades, the columns represent levels of experience, rather than specific grade levels; some students may only complete a foundational learning experience, whereas others may complete two, three, or even four experiences.

There are many opportunities for schools and students to create a pathway that makes the most sense for their unique interests, experiences, and resources. For example, after taking *Game Design & Digital Innovation*, a student may choose to take *Game Development* or *Application Development*, which flow most seamlessly from a content perspective. But a student may also decide that they prefer a breadth of experience and take another fundamentals course such as *Programming the Future*. Similarly, a student may have participated in an after-school robotics activity the summer after taking *Game Design & Digital Innovation* and determined that they are adequately prepared to take *Robotic Systems* as a follow-on course. These examples demonstrate the intended versatility and flexibility of the example pathways shown here.

Most schools will not be able to offer many options for specialized CS learning, so they may select a relevant subset of these pathways or substitute other areas of concentration. It may also be possible to offer some of these courses for dual credit with an agreement with an institution of higher education.

# Example Course Pathways



## Example Course Descriptions

### Computer Science Foundations

*Computer Science Foundations* supports all high school students, regardless of postsecondary goals, in developing the knowledge, skills, and dispositions necessary to navigate and understand the technology-driven world in which they live. Course content, organized into five Concepts (Algorithms & Design, Programming, Data & Analysis, Systems & Security, and Computing & Society) is integrated with four sets of Practices (Ethics & Social Responsibility, Computational Thinking, Inclusive Collaboration, and Human-Centered Design).

### Ethics of Computing

Technology is often considered an inherently neutral tool that has benefits and drawbacks that can be leveraged for better or for worse by the user or creator. However, this is a myth. Many scholars—including Ruha Benjamin, Safiya Noble, and Joy Buolamwini—have cataloged the ways in which computing technologies have embedded and extended biases. In *Ethics of Computing*, students explore the implications, and potential harm, for users and nonusers. Further, students consider how this knowledge translates into being a critical consumer and responsible creator of technology, weighing pros and cons and recognizing intended and unintended consequences. *(Note that offering this course is not a replacement for including ethics and impacts throughout all CS courses and instruction. This course provides an opportunity for deep, sustained, and focused learning specifically around ethics.)*

### Programming the Future

*Programming the Future* provides students who have a foundational understanding of computer science with an opportunity to explore various topics such as cybersecurity, artificial intelligence, and data science. While developing their programming skills, students will apply fundamental ideas in these areas to solve meaningful and interesting problems. *(Content covered in this course aligns with some of the Specialty I Standards across AI, Cybersecurity, Data Science, and Software Development.)*

### Game Design and Digital Innovation

*Game Design and Digital Innovation* is an ideal course for students who have a foundational understanding of computer science and a particular interest in applying that knowledge within the context of developing games or other 2D and 3D media, such as simulations. Students will learn aspects of the design process and leverage them in one or more projects of interest.

### Game Development

*Game Development* is ideal for students who have already taken *Game Design and Digital Innovation* and have an interest in bringing their designs to life. Students will engage in advanced study of programming and may apply those skills to create games and simulations in traditional, augmented reality (AR), virtual reality (VR), and/or extended reality (XR) environments. This course is intended to involve extensive collaboration through an intentional development process.

### Physical Computing

*Physical Computing* is a course for students who have a foundational understanding of CS and want to learn more about applying CS ideas to robots, sensors, and IoT devices. Students will use the engineering design process to address an individual/community need to solve an authentic problem.

### Robotic Systems

*Robotic Systems* is designed to be a follow-on course to *Physical Computing*. Students build upon existing knowledge of physical devices such as robots, sensors, and IoT devices in an effort to solve meaningful problems through thoughtful design and implementation processes.

### Information and Network Security

Innovations in artificial intelligence and quantum computing underscore the importance of securing information, programs, and applications for both personal and societal safety. *Information and Network Security* is intended to follow foundational programming and introductory cybersecurity learning experiences and prepare students for advanced study or workplace application of cybersecurity principles. The course involves learning about and applying security practices in authentic environments and contexts where possible.

Other titles may help capture student interest as well as reflect specific focal points and/or relevant current events. For example:

- *Game Design and Digital Innovation* → ***Digital Storytelling***
- *Physical Computing* → ***Digital Fashion***
- *Information and Network Security* → ***Hacking for Good***
- *Ethics of Computing* → ***Should We Ban TikTok?***

### AI and ML Programming

*AI and ML Programming* is intended to follow foundational programming and foundational AI learning experiences. Students will build upon this prerequisite knowledge to leverage AI in practical and innovative applications as well as to interrogate when opportunities to use AI may be unsafe or unreliable. *(This course includes a significant emphasis on data and should be paired with appropriate math learning.)*

### Data Science and Analytics

In a world that is increasingly informed and driven by data, it is necessary to understand data, where it comes from, how it is leveraged, and how it can impact life and work. *Data Science and Analytics* is a first in-depth course for students to investigate the various ways that data can be stored, accessed, modified, and visualized. Students will consider impacts and ethical considerations related to ownership and bias in data as well as how data visualizations can be misleading. While this course focuses on the computer science context, data science is increasingly interdisciplinary, and students will be afforded opportunities to apply analysis and visualization techniques in fields/topics of personal interest.

### Application Development

*Application Development* involves advanced study related to programming with a focus on developing mobile and desktop applications. Students will engage in collaborative development processes to solve a problem or address a personal or community need. In addition to development, students will test and refine their products to ensure usability and quality user experience, while also weighing ethical issues.

### Software Development

Software Development provides opportunities for extensive study in one or more programming languages, ideally that students have not experienced in previous coursework. Students learn about uses and advantages of particular programming languages and understand commonalities and differences across them. Students will engage in collaborative development processes to solve a problem or address a personal or community need using their programming skills. (*This course aligns with common first-year postsecondary programming courses—i.e., CS1, including AP CS A).*

### Computational Art

This course builds upon the student's previous experiences in computing and in art in order to provide opportunities for students to exercise their creativity and develop their portfolio in art by leveraging computing technologies. Content covered in this course may include:

- AI art generation
- Pixel-based art
- Ethical issues related to digital art
- Creative app development
- E-textiles
- Artistic applications of physical computing

### Computational Journalism

Designed for students who have completed a foundational computing course as well as a foundational journalism course, this class exposes students to computational techniques and issues related to journalism, including:

- Ethical issues, including data privacy and security
- Language processing and text analysis
- Reporting on technology and the industry
- Computing-based investigative techniques
- Data journalism, including data visualization

**A Note on Integrated Courses:** Integrated courses require collaborative, intentional planning to ensure balanced representation of all involved disciplines. Such planning is critical for integrated pathways (e.g., Computational Art, Computational Biology) as well as computing-intensive courses that rely heavily on other disciplinary content, such as data science. Curriculum planners and key support roles—including instructional coaches, language development specialists, and special education teachers—are also essential for designing courses that are accessible to all students. Because integrated courses can be difficult to fit into traditional high school department structures, we recommend carefully positioning them to ensure their intent is clear, all related disciplines are honored, and students understand how they align with their interests.

### Digital Humanities

This course, designed for students who have completed a foundational computing course as well as an introduction to the humanities, explores various techniques of digital humanities. Topics considered within the humanities are vast and thus, this course can be offered thematically. Students might, for example, develop digitized topographical maps to better understand historical battles (history lens), migration patterns (sociology lens), or artistic works (fine arts lens). Example topics include:

- Text mining and analysis, including via natural language processing
- Social network analysis
- Working with digital archives
- Digital mapping
- Audio, image, and video analysis
- New media studies, including software studies
- Ethical issues in the digital humanities

### Computational Biology

This course builds on students' previous experiences in an introduction to biology and a foundational computing course in order to develop knowledge and skills related to computational biology, including:

- Genetics and evolution
- Personalized medicine
- Digital pathology
- Data visualization
- Systems and networks in biology
- Algorithms in nature
- Ethical issues in computational biology

## Acknowledgements

This resource contains significant content from *Reimagining CS Pathways*, a community-wide effort led by CSTA and IACE alongside ACM, Code.org, College Board, CSforALL, and the ECEP Alliance. This work was supported by the National Science Foundation under Award No. 2311746. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

## Suggested Citation

Computer Science Teachers Association. (2026). *High school implementation guidance.* https://csteachers.org/pk12standards/resources

# Appendix: Broadening Participation in CS

Access to meaningful CS learning opportunities has historically been uneven, with many students—girls, Black, Latine, and Indigenous students, multilingual learners, students with disabilities, and students in rural or economically disadvantaged communities—facing barriers to participation. Closing that gap requires deliberate efforts to reach students who don't yet see themselves in computing.

## 1. Messaging

How CS is described shapes who considers it. Framing CS around social impact and interdisciplinary application, its intersections with medicine, art, and community problem-solving, resonates with students who might not respond to messaging focused on syntax or technical skill alone. Introductory courses should explicitly state that no prior experience is expected, to lower the intimidation barrier that keeps interested students from enrolling. The CAPE Framework (Capacity, Access, Participation, and Experience) offers a useful lens for auditing whether messaging and course design are reaching every student group or only some.

## 2. Personal Invitation

Direct, personal encouragement is one of the most effective recruitment tools available, particularly for students from underrepresented groups. Teachers, including those outside CS, can "tap" students who show relevant strengths in other classes: a specific, direct invitation ("I've seen how you solve problems; you have the mind for computer science") does more than a general announcement ever will.

## 3. Scheduling and Counseling

Systemic scheduling conflicts often block interested students before they ever choose against CS themselves. Schools should review whether introductory CS sections conflict with courses that have high enrollment from underrepresented groups, such as ESL blocks or specific athletic periods, and can run low-enrollment sections every other year rather than letting a single-section conflict clog the schedule indefinitely. Counselors for Computing (C4C) offers resources to help counselors advocate for CS as foundational literacy rather than a niche technical elective.

## 4. Peer Influence

Students often decide whether they belong in a space by looking at their peers. Recruiting within existing social groups, pairing older students as mentors to younger ones, and using current students to lead outreach events all build the social permission that formal recruitment alone cannot. Alumni from vocational, community college, or CTE programs can be more relatable role models for some students than four-year-college success stories.

## 5. Visible Belonging

What a classroom's walls display sends a signal before a teacher says a word. Representation across race, gender, culture, and career path, explicit belonging messages, and visible examples of current students' own work all shape whether a new student can imagine themselves succeeding in that room. Moving beyond images of isolated programmers at screens, toward collaboration, design, and real-world application, broadens who sees a place for themselves in the discipline. A simple audit, asking who is represented, who is missing, and whether a first-time student would feel welcomed, can surface what a classroom is signaling.

## Actionable Implementation Timeline

| Timeline | Action | Stakeholders |
|---|---|---|
| **Before recruitment season** | Refresh classroom visuals and materials to reflect student diversity and multiple pathways into computing | Teachers |
| **Months prior** | Set enrollment goals, convene a recruitment team, and present CS pathways at curriculum nights | Admin, Teachers, Counselors |
| **Weeks prior** | Conduct classroom visits and send personalized invitations to identified students | Teachers |
| **At registration** | Review enrollment demographics in real time and adjust scheduling to resolve conflicts | Admin, Registrar |
| **After enrollment** | Collect feedback from new students to refine future outreach | Teachers |

csteachers.org/pk12standards/resources