Revised PreK–12 Computer Science Standards

DRAFT 2.0 for Public Comment

SUGGESTED CITATION: Computer Science Teachers Association. (2025). *Revised preK–12 computer science standards: Draft 2.0 for public comment*. https://csteachers.org/k12standards/revision/

Table of Contents

Introduction

In a world increasingly powered by computing, students of all identities and chosen career paths need quality computer science (CS) education to become informed citizens and confident creators. Our vision for K–12 CS education is to ensure:

- All students are engaged and supported in learning CS, including its impacts on individuals, societies, cultures, democracies, and policies.
- Policies, pedagogies, and practices support all students learning CS.
- Standards align with the current and future needs for learning CS.

CSTA's K–12 Computer Science Standards delineate a core set of learning objectives designed to provide the foundation for a complete CS curriculum and its implementation at the K–12 level. The Standards, last published in 2017, were designed by educators to be coherent and comprehensible to teachers, administrators, and policymakers. The Standards are a primary resource for state and local education agencies when determining what K–12 students need to know and be able to do in CS. States widely adopted the Standards, which has impacted millions of students and promoted coherence in state policy, curriculum development, teacher certification, and teacher preparation and professional development across the U.S.

Increased implementation of CS in K–12 schools, continuously evolving research, and recent technological advancements necessitate an update to the CSTA K–12 Standards. As such, CSTA is conducting a three-year CSTA K–12 Standards revision process. Research began in fall 2023, writing kicked off in fall 2024, and the target date for publishing the updated standards is summer 2026.

CSTA defined the following values to guide the revision process and provide a lens for reflecting on and refining project outputs:

Equity-centered: Promotes broad and equitable access, participation, and experiences in CS education among all students.

Community-generated: Meets the needs of the community, including K–12 educators, postsecondary institutions, students, parents, and industry.

Future-oriented: Anticipates future needs of current learners, and prepares them for a future that is increasingly reliant on computing.

Grounded in research: Reflects the evolving body of knowledge of how students learn CS.

Flexible in implementation: Considers multiple pathways for meeting individual needs of learners, including regional, cultural, ability, social, and economic factors.

Ultimately, the goal of the CSTA K–12 Standards revision process is to *improve the quality of K–12 CS teaching and learning* by:

- Clarifying student learning outcomes across K–12 that:
  - Are driven by research and informed by teacher practice
  - Represent meaningful progressions
  - Define what is foundational for all students
- Maintaining coherence of instruction across the U.S. and beyond
- Ensuring that the updated standards work for teachers and students, regardless of the particular context or curriculum
- Providing flexibility and guidance for state and local adaptation

Additional information about the standards revision process can be found at https://csteachers.org/k12standards/revision/.

Additional Context for Reviewers

This document represents the second of three major drafts before CSTA publishes the updated standards in the summer of 2026. Feedback on this draft will inform how we refine the language and content of the standards, and how we develop clarification statements and other supports. At this stage, feedback that critiques the clarity and coherence of progressions within and/or across grade bands is most helpful. Feedback on specific word choices is less helpful now and will be more of a focus in the next round of feedback.

The writing team is currently using the following assumptions as guides throughout the writing process to help determine the volume, depth, and breadth of standards:

- Students will experience a certain amount of instructional time at each grade band:
    - » Elementary (Grades PreK–5): 20 to 40 hours per year (or 30 to 60 minutes per week)
    - » Middle school (Grades 6–8): the equivalent of one yearlong course
    - » High school (Grades 9–12): the equivalent of one yearlong course
- Implementation will vary and may include discrete courses and/or integration in other subject areas.
- Students will experience the full vertical progression (i.e., students learn the content in the PreK–5 standards before entering middle school and learn the content in the 6–8 standards prior to entering high school).

While these assumptions may not reflect the current reality of CS instruction in all schools, they represent a reasonable target for the future. We also acknowledge that the volume of standards in this current draft may not align with all of these assumptions and likely requires more instructional time than is feasible in many school contexts. We ask reviewers to focus on identifying which standards are essential and which are comparatively less important. This feedback will be invaluable as we reduce the number of standards in subsequent drafts.

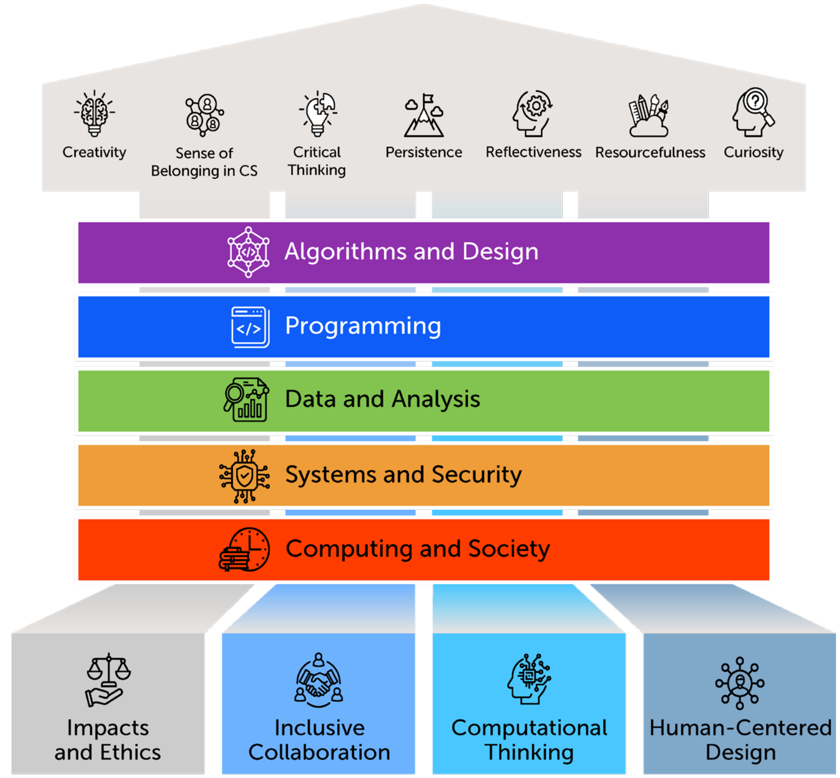
CSTA and the standards writing team look forward to reviewing your valuable insights as we work to define the future of PreK–12 CS education.

Navigating the Standards

We organized the foundational standards for PreK through high school around three primary components: Topic Areas, Pillars, and Dispositions. Topic Areas serve to organize standards by content. The five Topic Areas are: (1) Algorithms and Design, (2) Programming, (3) Data and Analysis, (4) Systems and Security, and (5) Computing and Society. We recognized artificial intelligence (AI) as a priority during the standards revision process. AI-related content is distributed across the five Topic Areas instead of being a discrete topic area.

Pillars consist of essential ideas and practices that cut across all topic areas. The four pillars are: (1) Impacts and Ethics, (2) Inclusive Collaboration, (3) Computational Thinking, and (4) Human-Centered Design.

Dispositions are habits of mind fostered within CS classrooms and developed through instruction that includes the topic areas and pillars. The highest-priority dispositions within the CS context are: creativity, sense of belonging in CS, critical thinking, persistence, reflectiveness, resourcefulness, and curiosity.



The draft PreK–12 standards within this organizational structure are foundational for all students. We adapted this structure from the [*Reimagining CS Pathways: High School and Beyond*] project, which aimed to create a community definition of what a foundational CS learning experience for all high school students includes and possible CS learning opportunities beyond that foundation. The graphic provides a visual representation of the relationship between topic areas, pillars, and dispositions.

The *Reimagining CS Pathways* project identified seven high school specialty areas for continued learning: (1) Software Development, (2) Cybersecurity, (3) Artificial Intelligence, (4) Physical Computing, (5) Data Science, (6) Game and Interactive Media Design, and (7) X + CS. We organized specialty areas in two levels: Specialty I and Specialty II. Specialty standards appear after the PreK–12 foundational standards. The draft specialty standards are intended for students who have completed foundational CS learning experiences and want to continue their study of CS in one or more of these areas. This likely includes students who wish to pursue computing-intensive postsecondary education or who have a particular interest in one or more specialty areas. Implementing the specialty standards may vary based on state/local contexts, resources, partnerships, and other priorities. Specialty I standards are foundational to the specialty area and Specialty II standards describe advanced study within the specialty area.

Lastly, throughout the standards, students attend to the societal and environmental impacts of computing. We use the term *societal* to encompass social, governmental, political, cultural, and economic factors.

Topic Areas

Algorithms and Design

Overview: An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. In early grades, students learn about age-appropriate algorithms from the real world. As they progress, students learn about the development, combination, and decomposition of algorithms; the evaluation of competing algorithms; and the difference between traditional algorithms and artificial intelligence/machine learning (AI/ML) algorithms.

The Algorithms and Design standards and the Programming standards are complementary and should be considered in tandem. Algorithms and Design standards focus more on program planning and evaluation, while Programming standards focus more on program implementation.

Subtopic	Overview
Algorithm Fundamentals	Designing algorithms, or step-by-step solutions to a task, are an essential component of CS. In early grades, students identify and create algorithms reflecting tasks in their daily lives. As students progress, they develop more complex algorithms, create visual representations of their solutions, and compare traditional algorithms to AI/ML algorithms.
Human-Centered Design	Many algorithms are designed to help people, so understanding user needs is essential to good algorithm design. In early grades, students identify how algorithms can support others and they use a human-centered approach to inform their choices when designing simple instructions. As students progress, they apply human-centered design principles to develop and refine computational algorithms, using feedback from others to improve performance and usability. In later grades, students evaluate algorithms for fairness, accessibility, and inclusivity, considering the broader impact of their designs on diverse user groups.
Problem Solving	While there may be many approaches to addressing a task, optimizing an algorithm can result in more efficient and accurate solutions. In early grades, students focus on patterns and decomposition to improve their algorithms. As students progress, they begin to evaluate the efficiency and accuracy of computational algorithms running under different conditions and use problem-solving skills to explore algorithms underlying opaque systems.
Impacts of Algorithms	Algorithms can have positive and negative effects on people and society. It is important to evaluate not only how well an algorithm works, but also who it benefits, who it may unintentionally harm, and why. In early grades, students begin by exploring how algorithms can lead to different results for themselves and others. As students progress, they learn to identify possible consequences of algorithmic decisions. In later grades, students critically analyze the societal impacts of algorithms, including issues of fairness, equity, accessibility, and bias, and consider how algorithmic systems can shape real-world experiences.

Programming

Overview: Programming controls all computing systems, empowering people to communicate with the world in new ways and solve compelling problems. The development process to create meaningful and efficient programs involves choosing which information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

The Algorithms and Design standards and the Programming standards are complementary and should be considered in tandem. Algorithms and Design standards focus more on program planning and evaluation, while Programming standards focus more on program implementation.

Notes:

- Many standards in this topic area discuss creating or reviewing "code." Not all code is text-based or screen-based. In particular, students in early grades may interface with tangible programming systems. Standards were written considering the programming tools commonly used at each grade level.
- While the other four topic areas contain a subtopic addressing societal and ethical impacts, the Programming topic area weaves societal and ethical impacts throughout its five subtopics.

Subtopic	Overview
Programming Fundamentals	Across grade levels, students focus on reading and interpreting code, translating algorithms into code, and understanding programming languages' types, syntax, and semantics. Although programming constructs repeat across this subtopic, their complexity is expected to increase as students advance through grade levels.
Data Handling	Understanding how programs structure and store data is necessary for successful programming. In early grades, students focus on identifying and labeling data in their daily lives and in age-appropriate programming languages. As students progress, they learn about and manipulate more complex data types. **Note:** This subtopic focuses on data used while programming. The Data and Analysis topic area focuses more on collecting, storing, and analyzing data with the use of computing.
Program Development	Programming involves paying attention to the organization and structure of code. In early grades, students strengthen their understanding of how to create programs. As students progress, they focus more on building on existing code, modularizing code, documenting code, and using AI tools to support their programming.
Testing and Refining Code	Ensuring that code works as intended is key to building reliable programs. In early grades, students focus on identifying and fixing errors in their programs. As students progress, they focus on more complex troubleshooting strategies, optimizing their code for efficiency and usability and assessing the accuracy and bias of AI-generated code.
Project Management	Programming projects are often collaborative. Collaboration is enhanced with defined project roles, shared documentation, and acknowledgment of project contributions. In the early grades, students focus on explaining how they completed their programming projects and how they collaborated effectively with others while coding. As they progress, students focus on program documentation, attribution, version control, and the computational and social impacts of programming decisions.

Data and Analysis

Overview: Computers collect and store data so it can be analyzed to better understand the world and make more accurate predictions. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important.

Subtopic	Overview
Data Fundamentals	Data is generated and collected by people, often using computing technologies such as sensors and other automated systems. Metadata is "data about data." Metadata provides context about data, including its origin, structure, and purpose. In early grades, students learn about different types of data and how data are generated, collected, and organized. As they progress, students gain experience with larger and more varied datasets, learn about more advanced data types and organization structures, and develop data documentation.
Data Processing	Computing devices process data to make it useful for analysis. In earlier grades, students learn how to use computational tools to manipulate data: filtering, grouping, summarizing, transforming, and reshaping data. As students progress, they apply computational methods to: automate data cleaning, identify and handle errors in data, and prepare data for analysis.
Data Investigation	Data investigations are a multistep process. When conducting data investigations, students pose data questions, use computational tools to collect and analyze data, create data visualizations, generate insights, and tell the story of their data. In early grades, students focus on asking simple questions that can be answered with small datasets. As students progress, they work with larger datasets, asking and answering more sophisticated questions that consider variability and relationships between multiple variables.
Impacts of Data Science	Students explore how data influences decision-making and impacts individuals and communities. Students examine the benefits, risks, and ethical considerations around data use. In early grades, students discuss how using data can help them make more informed decisions in their daily lives. As students progress, they explore issues related to bias in data, data privacy, artificial intelligence and machine learning, and large-scale societal and environmental impacts of data science applications.

Systems and Security

Overview: Systems and Security includes the broad categories of hardware and software, networks, and cybersecurity. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. Networks connect computing devices to share information and resources. Greater connectivity in the computing world has also led to an increased need for security to protect the information being transmitted.

Subtopic	Overview
Hardware and Software	Computing systems use hardware and software to communicate and process information in digital form. In early grades, students learn how systems use both hardware and software to represent and process information. As they progress, students gain a deeper understanding of the interactions between hardware and software at multiple levels within computing systems.
Networks	Computing devices communicate with one another across networks to share information. In early grades, students learn that computers connect them to other people, places, and things around the world. As they progress, students gain a deeper understanding of how information is sent and received across different types of networks.
Security	Transmitting information securely across networks requires appropriate protection. In early grades, students learn how to protect their personal information. As they progress, students learn about information transmission across devices, network design, and how to protect networks from different types of threats.
Impacts of Computing Systems	Humans created computing systems to accomplish tasks and solve problems. While there have been benefits, there have also been harms and the creation of new problems. In early grades, students examine the impacts of computing systems on individuals. As they progress, students learn about the impacts of computing systems on global society.

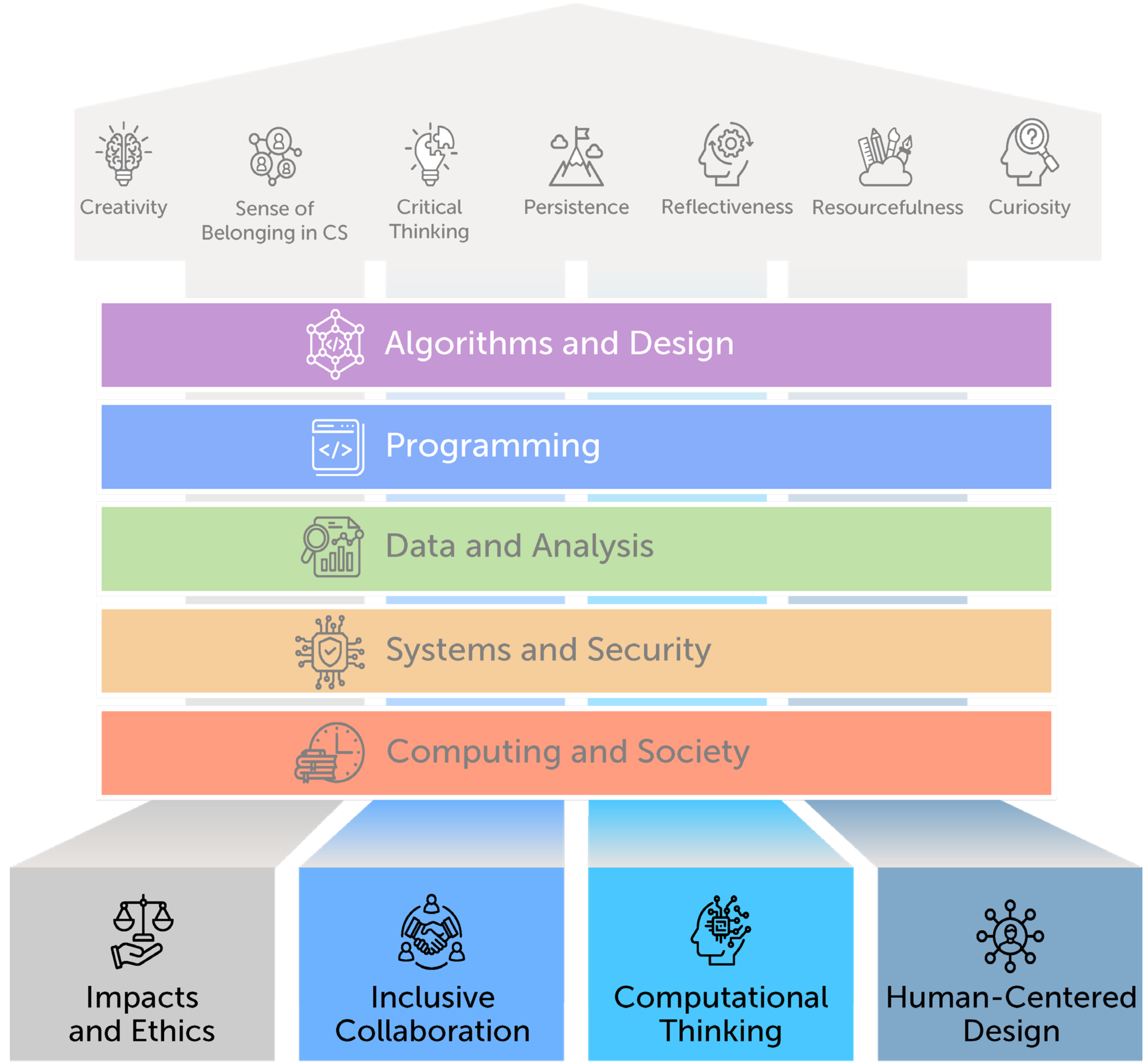
Computing and Society

Overview: Computing shapes—and is shaped by—individuals, communities, and cultures. It plays a powerful role in transforming daily life, economies, governments, and global systems. The impacts of computing are complex, producing both benefits and challenges, and raising important questions about equity, ethics, access, and responsibility. Students learn to understand, evaluate, and responsibly navigate the social implications of the digital world, including issues of equity, access, and the ethical use of technology. By examining the evolving relationship among computing, culture, and society, students become empowered to contribute thoughtfully and responsibly to a digital future.

Subtopic	Overview
History of Computing	Modern computing has roots in the 1800s and has evolved in response to human needs and innovations across time. In early grades, students first identify how computing is used in daily life. Then they focus on how technologies have evolved over time in response to social, scientific, and economic needs. In middle grades, students evaluate how historical challenges led to computing innovations and compare the roles played by individuals, communities, organizations, and governments in advancing these technologies. Students also explore the societal impacts of computing innovations. In later grades, students delve into the main eras of computing history and understand policy and legislation related to computing technologies. They also analyze the historic impacts of technologies, giving consideration to the factors that contributed to disparities across communities.
Emerging Technologies	Computing is a rapidly developing discipline. While other topic areas cover what students should know about the current field of CS, this subtopic focuses on recently developed technologies that have the potential to significantly impact society. In early grades, students learn how computing technology aids their daily life. They evaluate choices and consequences related to emerging technologies and identify problems that these advances could address. In middle grades, students evaluate how emerging technologies impact user experiences, while attending to ethical design principles. They explain how emerging technologies can inspire innovation and help people accomplish tasks in new ways. In later grades, students understand the core computational principles behind emerging technologies and compare the ethical considerations of emerging technologies, using various frameworks.
Career Exploration	Computing is foundational to nearly every career field. Understanding the role of computing in the workplace prepares students to make informed choices about their futures. In early grades, students recognize how digital tools and technologies support everyday work across professions. In middle grades, students explore how computational thinking drives innovation across industries and examine the ethical challenges that professionals may encounter. In later grades, students connect computing to their personal interests and career goals, investigate computing-related pathways, and analyze how advancements in technology foster new opportunities for growth across diverse fields.

Pillars

Pillars differ from Topic Areas and include key ideas and practices that are integral to each Topic Area. All Pillars contain practices. Each standard is intended to reflect both content and one or more practices. Two Pillars, Impacts and Ethics and Human-Centered Design, also contain key ideas. These key ideas are intended to appear in relevant Topic Areas as one or more distinct subtopics.



Impacts and Ethics

The goal of the Impacts and Ethics pillar is for students to understand the societal and ethical implications of pervasive computing technologies. Students should study these implications to support themselves in becoming responsible creators of technology who use computing to benefit all members of society, including the most vulnerable. The key ideas for Impacts and Ethics are synthesized from multiple sources, including the Standards for Technological and Engineering Literacy (STEL; ITEEA, 2020), the National Curriculum Standards for Social Studies (NCSS, 2010), and the K–12 CS Framework (2016). The practices for Impacts and Ethics are based on the Association for Computing Machinery's list of general ethical principles (ACM, 2018).

Key Ideas

1. **Access to and use of computing technologies differ across groups of people** (e.g., by socioeconomic status; geographic location; cultural background; differences in ability, gender, race and ethnicity, and other social identities).
2. **Computing has both positive and negative impacts on society and the environment.**
   - **a.** Computing technologies impact people differently depending on their values, socioeconomic statuses, geographic locations, cultural backgrounds, abilities, and social identities.
   - **b.** There are laws to regulate the design and use of computing technologies, as well as unsuccessful attempts to enact such laws.
3. **Responsible creation and use of computing technology requires attention to ethical issues.**
   - **a.** Decisions made about the design and use of computing technologies involve consideration of costs, benefits, and trade-offs.
   - **b.** Computing technology should be optimized to perform the task for which it was designed with the fewest possible harmful impacts on users and the environment.

Practices

1. **Use computing for the public good.**
   - **a.** Create computing technologies that are inclusive and accessible for diverse groups of people. Do not create technologies where the benefits and harms accrue to different groups of people. Harms can include physical or mental injury, destruction or disclosure of information, and damage to property, reputation, and the environment.
   - **b.** Carefully consider the potential impacts of new computing technologies to avoid unintentionally causing harm. Avoid, mitigate, and remediate harms caused by computing technologies.
2. **Respect users and other creators when creating computational technologies.**
   - **a.** Respect the work required to produce new ideas, inventions, creative works, and computing artifacts. Only use others' work with permission and give appropriate attribution to others' work.
   - **b.** Respect users' privacy. Clearly explain to users what information is being collected, how their information will be stored and used, and when it will be disposed of. Only collect the minimum amount of information necessary and protect data from unauthorized access and accidental disclosure.

Inclusive Collaboration

The core of the Inclusive Collaboration Pillar is to help students develop productive collaborations with diverse groups of people. The practices in this Pillar, which were synthesized from the STEL (ITEEA, 2020), the Social Justice Standards (Learning for Justice, n.d.), the Framework for 21st Century Learning (Partnership for 21st Century Skills, 2009), and the K–12 CS Framework (2016), address communication and other social skills, project management skills, and personal conduct when working with others.

Practices

1. **Communicate clearly, using oral, written, and nonverbal methods, and practice intercultural competence.**
   - **a.** Cultivate working relationships with individuals possessing diverse perspectives, skills, and personalities. Aim to build empathy, respect, understanding, and connection.
   - **b.** Communication is an exchange; students should know when it is appropriate to speak and when it is appropriate to listen. Understand that different groups of people have different norms around communication. Express curiosity about the perspectives of others and exchange ideas in an open-minded way.
   - **c.** Develop conflict resolution skills, including active listening, empathy, perspective-taking, and emotional management. These skills help students solve problems and find common ground with their peers, leading to more effective collaborations.
   - **d.** Leverage social and cultural differences to create new ideas and increase both innovation and the quality of work.
2. **Actively manage projects and project teams.**
   - **a.** Manage time effectively. Break a project into smaller parts. Work backward from major deadlines and set interim deadlines for each part of the project, considering whether some parts need to be completed before other parts.
   - **b.** Document workflows, work processes, and work products to create a shared understanding among team members and transparency for those with interests in the project work.
   - **c.** Establish common goals and team norms. Define team roles and create equitable workloads. Leverage team members' strengths to accomplish common goals and value the individual contributions of each team member. Be flexible, helpful, and willing to make necessary compromises to accomplish a common goal.
3. **Act responsibly and demonstrate accountability to others.**
   - **a.** Be honest and reliable. Actively participate in collaborative work and assume shared responsibility for work products. Be on time for team meetings and meet team deadlines.
   - **b.** Solicit and incorporate feedback from team members and other interested parties.
   - **c.** Engage in self-reflection. Students should continually examine their own actions, decisions, and interactions with team members during a project to learn from experience and identify areas needing improvement.

Computational Thinking

Computational thinking is a way of thinking about problems and formulating problems and solutions so that an information-processing agent (e.g., a computer) can solve them. Computational thinking practices should underpin instruction in each Topic Area and connect students' CS learning experiences. Embedded within this Pillar is the engineering design process, in which students identify and define computational problems, develop computational solutions, and iteratively test, refine, and optimize those solutions. These practices are largely based on the original Computational Thinking practices from the K–12 CS Framework (2016), but also incorporate ideas from the Common Core State Standards for Mathematical Practice (NGA Center & CCSSO, 2010, pp. 6–8).

Practices

1. **Define and decompose computational problems.**
   - **a.** Identify complex, interdisciplinary, real-world problems that can be solved computationally.
   - **b.** Evaluate whether it is appropriate and feasible to solve a problem computationally.
   - **c.** Decompose complex problems into manageable subproblems that could integrate existing solutions or procedures.
   - **d.** Analyze computational problems, identify sources of error, and plan solution pathways.
   - **e.** Identify and fix errors in computer programs and computing systems, using a systematic process.
2. **Recognize patterns; develop and use algorithms and abstractions.**
   - **a.** Look for patterns and repetition. Identify and create general methods (i.e., algorithms) for performing repetitive tasks.
   - **b.** Extract common features from a set of integrated processes or complex phenomena.
   - **c.** Evaluate existing technological functionalities and incorporate them into new designs.
   - **d.** Create modules and develop points of interaction that can apply to multiple situations and reduce complexity.
   - **e.** Model phenomena and processes and simulate systems to understand and evaluate potential outcomes.
3. **Create and/or modify computational artifacts, using an engineering design process.** The artifacts can be for practical uses, entertainment, or personal expression.
   - **a.** Plan the development/modification of computational artifacts, using an iterative process that includes reflection on and modification of the plan, taking into account key features, time and resource constraints, and user expectations.
   - **b.** Put the plan into action to develop the artifact. For group development projects, apply the principles of inclusive collaboration.
   - **c.** Iteratively test, evaluate, and refine computational artifacts to enhance performance, reliability, usability, and accessibility.

Human-Centered Design

Using human-centered design practices is a critical piece of responsibly creating computational solutions. Human-centered design includes principles of human-computer interaction. The following key ideas and practices are drawn from a variety of well-known sources on human-centered design, including the National Institute of Standards and Technology (NIST, 2021), the Interaction Design Foundation (IDF, n.d.a, n.d.b), and the UX Design Institute (Vinney, 2023), as well as the K–12 CS Framework (2016).

Key Ideas

1. **Human-centered design is an approach to designing products and systems that focuses on human needs and requirements.** This approach improves user satisfaction, accessibility and sustainability, and human well-being. Human-centered design also helps counteract or mitigate ways that products and systems may harm humans.
2. **Humans vary in our physical characteristics and abilities, social and cultural backgrounds, and attitudes and motivations.** There is no such thing as a single "average user" of computational technology, and designers of computational technologies should not assume their own experiences are universal. Instead, designers should work to understand how human variation relates to computational products and systems and purposely design to accommodate this variation.
3. **Humans are capable of complex reasoning, but we also have limitations to our attention and memory and biases toward our own experiences and beliefs.** Designers of computational technologies should understand how human cognition influences the way we interact with and process information from computational technologies.
4. **Humanity-centered design focuses on designing products and systems that consider human society, human communities, and the environment.** Rather than focusing on the experiences of individual users, humanity-centered design focuses on the interconnectedness of the world and complex systems of cause-and-effect relationships to address larger problems.

Practices

1. **Be human-centered.** As a part of any design and development process, students should learn about the people who may use or otherwise be affected by a computational technology. When possible, students should include a diverse sample of people to inform the design and development of computational technologies, including identifying what problems need to be solved, evaluating potential solutions, and testing products in development.
2. **Start with small and simple interventions and iterate** rather than rushing into big design solutions. Continually prototype, test, review, and refine solutions to make sure they meet human needs without causing harms.

Foundational Standards for PreK–12

Naming Conventions for Foundational Standards

Each of the identifiers for foundational standards follows this naming convention:

Grade band		Topic area		Subtopic		Number
XX	-	YYY	-	ZZ	-	##

There are standards for each individual elementary grade. All identifiers for elementary standards begin with an E, followed by a letter (K to indicate PreK–Kindergarten standards) or number (1–5) to indicate the grade level. Standards for grades 6–8 are banded together as middle school standards. Identifiers for middle school standards begin with "MS." Standards for grades 9–12 are banded together as high school standards. Identifiers for high school standards begin with "HS."

The next two groups of characters indicate the topic area and subtopic for each standard. The following table shows the abbreviations for each topic area and subtopic. The last two digits of each standard reflect the standard number. The foundational standards begin with 01 for each topic area in each grade or grade band. The standards are numbered continuously across subtopic areas within each grade or grade band. The numbering restarts for each topic area.

Topic Area Abb	Topic Area	Subtopic Abb	Subtopic
ALG	Algorithms and Design	**AF**	Algorithm Fundamentals
		HD	Human-Centered Design
		PS	Problem Solving
		IM	Impacts of Algorithms
PRO	Programming	**PF**	Programming Fundamentals
		DH	Data Handling
		PD	Program Development
		TR	Testing and Refining Code
		PM	Project Management
DAA	Data and Analysis	**DF**	Data Fundamentals
		DP	Data Processing
		DI	Data Investigations
		IM	Impacts of Data Science
SAS	Systems and Security	**HW**	Hardware and Software
		NW	Networks
		SC	Security
		IM	Impacts of Computing Systems
CAS	Computing and Society	**HC**	History of Computing
		ET	Emerging Technologies
		CE	Career Exploration

Algorithms and Design

Algorithm Fundamentals

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-ALG-AF-01.** Identify algorithms in daily activities.	**MS-ALG-AF-01.** Optimize visual representations of algorithms. **MS-ALG-AF-02.** Describe how data is taken in (as input), stored, processed, and then produced as a result (as output) in a computational solution.	**HS-ALG-AF-01.** Develop algorithms that include variables, data, and storage, using authentic, real-world data. **HS-ALG-AF-02.** Describe the differences between deterministic algorithms and probabilistic algorithms. **HS-ALG-AF-03.** Justify the use of sequence, selection, or iteration while creating an algorithm. **HS-ALG-AF-04.** Design procedures that include sequence, selection, and iteration. **HS-ALG-AF-05.** Model a computational algorithm, using a flowchart.
1	**E1-ALG-AF-01.** Create algorithms that include step-by-step instructions to complete a daily task.		
2	**E2-ALG-AF-01.** Model daily processes by creating and following algorithms that include sequence, events, and iteration to complete tasks.		
3	**E3-ALG-AF-01.** Write the steps in algorithms that include sequence, events, iteration, and selection to complete a task or solve a problem, using everyday language.		
4	**E4-ALG-AF-01.** Create visual representations for algorithms that include sequence, events, iteration, and selection to solve a problem or complete a task.		
5	**E5-ALG-AF-01.** Create visual or textual representations of algorithms that include sequence, events, iteration, selection, and variables to solve a problem or complete a task.		

Human-Centered Design

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-ALG-HD-02.** Identify ways that technology might help others.	**MS-ALG-HD-03.** Design algorithms, using human-centered design principles such as empathy, user needs and requirements, and accessibility. **MS-ALG-HD-04.** Refine algorithms iteratively through user feedback to improve usability, accessibility, and user experience.	**HS-ALG-HD-06.** Design algorithms for diverse audiences by incorporating feedback, evaluating effectiveness, and identifying potential harms or unintended consequences.
1	**E1-ALG-HD-02.** Design an improvement to a technology to solve a problem for someone else.		
2	**E2-ALG-HD-02.** Discuss how human problems might be solved with the assistance of algorithms or programs.		
3	**E3-ALG-HD-02.** Design an algorithm or program that solves a problem for and meets the needs of someone else.		
4	**E4-ALG-HD-02.** Modify an algorithm or program, based on feedback, to better meet the needs and requirements of others.		
5	**E5-ALG-HD-02.** Develop an algorithm to solve a problem, using a process that considers the needs, requirements, and feedback of others.		

Problem Solving

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-ALG-PS-03.** Identify the individual parts that make up a whole object (e.g., a car has a body, windshield, doors, and wheels).	**MS-ALG-PS-05.** Demonstrate the correctness of algorithms for given inputs. **MS-ALG-PS-06.** Verify whether an algorithm can or cannot help solve a problem. **MS-ALG-PS-07.** Hypothesize about internal processes and functions of opaque (black-box) systems to explain how outputs are produced.	**HS-ALG-PS-07.** Determine how variables and constants affect an algorithm's results. **HS-ALG-PS-08.** Evaluate algorithms for efficiency, correctness, and clarity, using defined metrics or test cases. **HS-ALG-PS-09.** Optimize the design of algorithmic solutions, using abstractions such as procedures, modules, lists, and/or objects. **HS-ALG-PS-10.** Contrast different types of probabilistic algorithms and models, noting the strengths and limitations of their reasoning.
1	**E1-ALG-PS-03.** Describe the function of individual parts within a whole object or system.		
2	**E2-ALG-PS-03.** Modify algorithms with repeating patterns to use iteration instead of repeated instructions.		
3	**E3-ALG-PS-03.** Decompose a problem or task into smaller components to develop an algorithm.		
4	**E4-ALG-PS-03.** Assess an algorithm's effectiveness in solving a problem.		
5	**E5-ALG-PS-03.** Analyze algorithms for efficiency and accuracy, improving them with iteration and selection where appropriate.		

Impacts of Algorithms

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-ALG-IM-04.** Identify how algorithms might influence everyday activities and outcomes.	**MS-ALG-IM-08.** Describe common societal impacts, ethical issues, and biases of deterministic and probabilistic algorithms. **MS-ALG-IM-09.** Modify an algorithm to address a specific societal impact, ethical issue, or bias.	**HS-ALG-IM-11.** Evaluate the societal impacts, ethical implications, potential biases, and unintended consequences of both deterministic and probabilistic algorithms.
1	**E1-ALG-IM-04.** Identify how changes to algorithms might lead to different outcomes.		
2	**E2-ALG-IM-04.** Compare how different algorithms for solving the same problem may affect people differently.		
3	**E3-ALG-IM-04.** Describe how algorithms might impact peers in varied situations.		
4	**E4-ALG-IM-04.** Evaluate how different algorithms may affect outcomes, situations, and people with a wide range of needs.		
5	**E5-ALG-IM-04.** Examine different perspectives, abilities, and points of view when designing algorithms and programs.		

Programming

Programming Fundamentals

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-PRO-PF-01.** Choose the code that accurately matches an algorithm that includes sequence. **EK-PRO-PF-02.** Create programs that include a sequence to complete a task.	**MS-PRO-PF-01.** Analyze how a segment of code works by identifying and describing the roles of key components (e.g., variables, conditionals, loops, functions). **MS-PRO-PF-02.** Develop code from algorithms that include variables, data, and storage. **MS-PRO-PF-03.** Compare how programming languages with differing syntaxes or semantics influence the coding process and problem-solving strategies.	**HS-PRO-PF-01.** Convert an algorithm written in pseudocode into a program that uses sequence, selection, iteration, procedures with parameters, and lists. **HS-PRO-PF-02.** Analyze the purpose of a segment of code.
1	**E1-PRO-PF-01.** Create code from an algorithm that includes sequence and events to express ideas or complete a task.		
2	**E2-PRO-PF-01.** Create code from an algorithm that includes sequence, events, and iteration to express ideas or complete a task.		
3	**E3-PRO-PF-01.** Develop code from a student-created algorithm that includes sequence, events, iteration, and selection to express ideas or complete a task.		
4	**E4-PRO-PF-01.** Develop code from a student-created algorithm that includes sequence, events, iteration, and selection to express ideas or complete a task.		
5	**E5-PRO-PF-01.** Develop code from student-created algorithms that include sequence, events, iteration, selection, and variables to express ideas, complete a task, or solve a problem.		

Data Handling

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-PRO-DH-03.** Identify gestures and symbols (e.g., restroom signs, arrows for directions, thumbs up) in everyday life that represent information used to make binary choices.	**MS-PRO-DH-04.** Represent data, using appropriate data structures, including variables and collection types. **MS-PRO-DH-05.** Use appropriate data types (e.g., integers, text, Boolean) to store, update, and evaluate data within a program. **MS-PRO-DH-06.** Use iteration (e.g., loops) to access, update, and process elements in a collection.	**HS-PRO-DH-03.** Perform operations on collections of data of the same type, using appropriate structures (e.g., lists, tables, or dictionaries) to solve problems. **HS-PRO-DH-04.** Create programs that use collections to generalize solutions instead of repeatedly using simple variables. **HS-PRO-DH-05.** Compare and contrast fundamental data types (e.g., integer, text, float) and their uses.
1	**E1-PRO-DH-02.** Identify terms that refer to values that change over time in everyday life (e.g., today's date).		
2	**E2-PRO-DH-02.** Label different representations of information (e.g., day of the week, line leader, your birthday) with a name and whether its value is constant or changes.		
3	**E3-PRO-DH-02.** Identify the variables being stored and manipulated in a program.		
4	**E4-PRO-DH-02.** Trace how data flows through and alters values in an existing program, using variables.		
5	**E5-PRO-DH-02.** Use different types of variables to store, compare, and modify data, based on user input or program logic.		

Program Development

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	—	**MS-PRO-PD-07.** Modify existing programs that incorporate sequence, selection, and iteration. **MS-PRO-PD-08.** Create modular programs that incorporate sequence, selection, and iteration. **MS-PRO-PD-09.** Utilize reference documentation, online resources, and programming tools to assist in writing, debugging, and improving code. **MS-PRO-PD-10.** Simulate program development roleplay to evaluate team decisions, considering potential harms such as data privacy, bias, or negative societal impacts.	**HS-PRO-PD-06.** Assess opportunities to incorporate external code, utilizing documentation, libraries, APIs, development tools, and online resources. **HS-PRO-PD-07.** Evaluate the societal impacts, ethical implications, potential biases, and unintended consequences of using AI tools for program development. **HS-PRO-PD-08**. Create modular programs that incorporate sequence, nested selection, nested iteration, and lists. **HS-PRO-PD-09.** Adapt programs to improve modularity, using constructs such as procedures, modules, and/or objects.
1	—		
2	—		
3	—		
4	—		
5	—		

Testing and Refining Code

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-PRO-TR-04.** Identify errors in programs that do not work as expected.	**MS-PRO-TR-11.** Use standard practices to test, debug, document, and peer-review code. **MS-PRO-TR-12.** Modify a program to improve usability and accessibility. **MS-PRO-TR-13.** Describe issues of accuracy and bias in computer-generated code.	**HS-PRO-TR-10.** Modify a program to improve or change functionality, usability, accessibility, safety, accuracy, or inclusivity of a program. **HS-PRO-TR-11.** Argue how a program does and does not address a given problem. **HS-PRO-TR-12.** Verify that a program performs according to its design specifications and documented test cases. **HS-PRO-TR-13.** Assess the accuracy, efficiency, and ethical considerations of computer-generated code.
1	**E1-PRO-TR-03.** Debug a program that includes sequence and events to correct errors.		
2	**E2-PRO-TR-03.** Debug errors in programs that include sequence, events, and iteration.		
3	**E3-PRO-TR-03.** Debug errors in iteration or selection in a program. **E3-PRO-TR-04.** Create alternative versions of a program to solve the same problem or complete the same task.		
4	**E4-PRO-TR-03.** Debug errors in a program that includes sequence, events, iteration, and selection. **E4-PRO-TR-04.** Compare programs that complete a similar task and determine which would be easiest to repurpose.		
5	**E5-PRO-TR-03.** Debug programs, using systematic strategies (e.g., testing while writing, tracing code) to ensure they run as intended. **E5-PRO-TR-04.** Create a unique program by modifying other programs or incorporating portions of other programs into one's own work to develop something new or add more advanced features.		

Project Management

<table>
<tr><th colspan="2">Grades PreK–5</th><th>Grades 6–8</th><th>Grades 9–12 Foundations</th></tr>
<tr><td>PreK–K</td><td>EK-PRO-PM-05. Articulate how one has completed a task or solved a problem that takes multiple steps.

EK-PRO-PM-06. Create a programming sequence to complete a task, working with a partner.</td><td rowspan="6">MS-PRO-PM-14. Explain the importance of attribution and intellectual property in programming.

MS-PRO-PM-15. Apply inclusive collaboration practices to support all stages of programming, from planning to testing.

MS-PRO-PM-16. Document a program, using comments, descriptive names, and structured guides, to improve readability, enable collaboration, and explain complex logic or code intent.</td><td rowspan="6">HS-PRO-PM-14. Apply correct attribution to intellectual property (e.g., code, libraries, use of AI).

HS-PRO-PM-15. Apply version control to the process of program development.

HS-PRO-PM-16. Critique the societal impacts and unintended computational impacts of program modifications from the lenses of specific user groups.

HS-PRO-PM-17. Organize a project workflow with a design document that outlines individual responsibilities and task dependencies.</td></tr>
<tr><td>1</td><td>E1-PRO-PM-04. Explain the function of a code segment.

E1-PRO-PM-05. Collaborate with a partner to develop a program that expresses an idea or solves a problem.</td></tr>
<tr><td>2</td><td>E2-PRO-PM-04. Document the steps taken and choices made during program development, recognizing the contributions of others in the process.

E2-PRO-PM-05. Use structured constructive feedback from a peer to improve a programming project.</td></tr>
<tr><td>3</td><td>E3-PRO-PM-05. Articulate the function of a specific aspect of a program.

E3-PRO-PM-06. Collaborate with a team to create a program, ensuring that all team members have a role and contribute equally.</td></tr>
<tr><td>4</td><td>E4-PRO-PM-05. Document a program, using embedded or external comments, to clarify its functions for others.

E4-PRO-PM-06. Construct individual components of a program that are collaboratively assembled into a working project.</td></tr>
<tr><td>5</td><td>E5-PRO-PM-05. Create embedded or external documentation of a programming project, including explanations of code functionality, choices made, and attribution for the contributions of others.

E5-PRO-PM-06. Collaborate with peers to design, implement, document, and review a programming project that expresses an idea or solves a problem.</td></tr>
</table>

Data and Analysis

Data Fundamentals

Grade	Grades PreK–5	Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-DAA-DF-01.** Demonstrate how people create and collect various types of data to help answer questions.	**MS-DAA-DF-01.** Examine the relationship between data and metadata collected using computational tools. **MS-DAA-DF-02.** Evaluate the impact of precision and granularity in data collection and analysis, considering how different levels of detail affect accuracy, storage, and interpretation (e.g., rounding numerical values, image resolution, or sensor sampling rates). **MS-DAA-DF-03.** Use computational tools such as spreadsheets to collect and organize quantitative and qualitative data. **MS-DAA-DF-04.** Create metadata that generally describes the purpose and contents of datasets, who collected the data, and how the data were collected.	**HS-DAA-DF-01.** Describe the differences among nominal, ordinal, discrete, and continuous data and how each type of data might be generated and used in data analysis. **HS-DAA-DF-02.** Use computational tools such as programs or spreadsheets to generate data that fits certain parameters (e.g., random numbers, normal distributions, random samples from a larger dataset) for use in simulations. **HS-DAA-DF-03.** Explain the differences between ways of organizing and storing data (e.g., tall vs. wide formats) and when to use them. **HS-DAA-DF-04.** Create a data dictionary that describes the names and types of attributes, allowable values/ranges for each attribute, and logical relationships between variables in a dataset.
1	**E1-DAA-DF-01.** Compare numeric and non-numeric data in terms of how they are collected and what they can tell us.		
2	**E2-DAA-DF-01.** Collect numeric and non-numeric data, using multiple methods, including observation, measurement, and survey.		
3	**E3-DAA-DF-01.** Collaborate with peers to collect numeric and non-numeric data, following consistent procedures. **E3-DAA-DF-02.** Evaluate collected data for accuracy and completeness.		
4	**E4-DAA-DF-01.** Organize data into tables where rows represent a “record” (e.g., a survey respondent) and columns represent attributes (e.g., case identifiers and the responses given by each person).		
5	**E5-DAA-DF-01.** Collect different types of data, using computational tools (e.g., sensors, cameras, online surveys).		

Data Processing

<table>
<tr><th colspan="2">Grades PreK–5</th><th>Grades 6–8</th><th>Grades 9–12 Foundations</th></tr>
<tr><td>PreK–K</td><td>—</td><td rowspan="6">MS-DAA-DP-05. Use computational tools to organize, filter, group, and aggregate data (e.g., calculating frequencies, measures of center, and measures of spread).

MS-DAA-DP-06. Use computational tools to manipulate data (e.g., calculating new variables or attributes from existing ones).

MS-DAA-DP-07. Identify errors in data (e.g., missing values, incorrectly formatted dates) and ways to fix them.</td><td rowspan="6">HS-DAA-DP-05. Reshape data, using computational tools (e.g., transform from tall to wide format, make or flatten hierarchy).

HS-DAA-DP-06. Use computational tools to clean and organize text-based data (e.g., trimming white space, standardizing capitalization, and sorting open-ended responses into categories).

HS-DAA-DP-07. Use computational tools to identify and address outliers and out-of-bounds values in data.

HS-DAA-DP-08. Evaluate different approaches to handling missing data (e.g., converting “N/A” to blank cells or replacing missing data with 0s).

HS-DAA-DP-09. Evaluate data quality by verifying logical consistency and compliance with expected data types and ranges.</td></tr>
<tr><td>1</td><td>—</td></tr>
<tr><td>2</td><td>—</td></tr>
<tr><td>3</td><td>—</td></tr>
<tr><td>4</td><td>—</td></tr>
<tr><td>5</td><td>—</td></tr>
</table>

Data Investigation

<table>
<tr><th colspan="2">Grades PreK–5</th><th>Grades 6–8</th><th>Grades 9–12 Foundations</th></tr>
<tr><td>PreK–K</td><td>EK-DAA-DI-02. Compare questions that can be answered with data investigations and questions that are answered through other means.

EK-DAA-DI-03. Recognize patterns that people and machines can use to make decisions.</td><td rowspan="6">MS-DAA-DI-08. Pose data questions that anticipate variability in the data.

MS-DAA-DI-09. Use computational tools to identify relationships among variables in a dataset and make classifications or predictions.

MS-DAA-DI-10. Create data visualizations to demonstrate how different design choices can affect a data visualization’s clarity, visual appeal, accessibility, and capacity to accurately communicate insights from data investigations.

MS-DAA-DI-11. Evaluate the quality and limitations of a dataset for answering different data questions.

MS-DAA-DI-12. Share the story of a data investigation with peers, including what data question was asked, how the data was collected and analyzed, and what evidence supports the conclusion.</td><td rowspan="6">HS-DAA-DI-10. Pose data questions involving multiple variables, using computationally collected data.

HS-DAA-DI-11. Design a process that uses computational tools to collect data to answer a question, make classifications, or make predictions.

HS-DAA-DI-12. Evaluate the results of data simulations to help answer data questions and inform decision-making.

HS-DAA-DI-13. Evaluate data visualizations, based on visual appeal, clarity, accessibility, and integrity.

HS-DAA-DI-14. Justify a data-driven conclusion to a data question by creating a formal report that explains the investigative process and supports the conclusion with evidence, including potential biases or limitations that might lead to alternative interpretations.</td></tr>
<tr><td>1</td><td>E1-DAA-DI-02. Investigate questions that can be answered by manually collecting data in students’ everyday environments.

E1-DAA-DI-03. Explore how patterns can be used by people and machines to make predictions and classify objects into categories.</td></tr>
<tr><td>2</td><td>E2-DAA-DI-02. Evaluate different representations of the same data for accuracy, clarity, and accessibility.

E2-DAA-DI-03. Explain how computing technologies can learn from patterns in data.</td></tr>
<tr><td>3</td><td>E3-DAA-DI-03. Create a data visualization and a brief narrative to report the process and results of a data investigation.

E3-DAA-DI-04. Explain how AI models can evolve when new data is added to the training set.</td></tr>
<tr><td>4</td><td>E4-DAA-DI-02. Investigate a data question involving relationships between multiple attributes.

E4-DAA-DI-03. Explore the relationship between the properties of training data (e.g., size, features, biases) and an AI model’s output.</td></tr>
<tr><td>5</td><td>E5-DAA-DI-02. Analyze a dataset to identify the nature and possible sources of variability in the data.

E5-DAA-DI-03. Use an AI learning platform to train an AI model to make decisions based on inputs from a dataset.</td></tr>
</table>

Impacts of Data Science

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-DAA-IM-04.** Explore how data can help a person make informed decisions in everyday life.	**MS-DAA-IM-13.** Explain the benefits and risks of allowing personal data to be collected and incorporated into datasets. **MS-DAA-IM-14.** Analyze how decisions made during data collection, data processing, data analysis, and data presentation can lead to biased data, misleading conclusions, and compromised AI models. **MS-DAA-IM-15.** Analyze the societal impacts of data-driven algorithms and computational systems, including AI.	**HS-DAA-IM-15.** Analyze the consequences of using data in AI/ML applications, including how biased training data can lead to biased output and reinforce societal inequalities and injustices with misinformation and disinformation. **HS-DAA-IM-16.** Evaluate the societal, environmental, and ethical implications of large-scale data collection and usage. **HS-DAA-IM-17.** Compare and contrast efforts to ensure responsible data usage through policies and regulations. **HS-DAA-IM-18.** Write plans to solve a specific problem, using the results of a data investigation.
1	**E1-DAA-IM-04.** Investigate a variety of data questions that address the needs of a person or community (e.g., classroom, family, school).		
2	**E2-DAA-IM-04.** Distinguish among data collection approaches that may lead to poor information.		
3	**E3-DAA-IM-05.** Design a data collection approach that addresses the needs of people from different backgrounds or groups.		
4	**E4-DAA-IM-04.** Analyze privacy concerns related to collected data.		
5	**E5-DAA-IM-04.** Analyze the benefits and risks of using data in real-world scenarios, including AI.		

Systems and Security

Hardware and Software

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-SAS-HW-01.** Examine the use of tools (e.g., tablets, robots, rulers, pencils/ erasers, tweezers, magnifying glasses, mobile devices, apps) to accomplish tasks and/or solve problems.	**MS-SAS-HW-01.** Describe the structure and organization of file systems, including file naming conventions, directories, and file permissions, and their role in storing and managing data.	**HS-SAS-HW-01.** Describe computing devices used in industry (e.g., robots, electronic control units in vehicles, medical imaging devices), how they function, and how they are used to accomplish tasks and/or solve problems.
1	**E1-SAS-HW-01.** Describe the purposes of basic hardware components (e.g., display, system unit, keyboard and other peripherals) in a computing system, using accurate terminology.	**MS-SAS-HW-02.** Differentiate between web-based applications and local software, based on user needs, systems requirements, and potential societal, environmental, and ethical impacts.	**HS-SAS-HW-02.** Differentiate operating systems as a special type of software that manages both the hardware and other software components of a computing system, including handling memory and peripherals.
2	**E2-SAS-HW-01.** Justify the use of software and its functions to accomplish tasks and/or solve problems.	**MS-SAS-HW-03.** Apply basic troubleshooting processes to identify and fix common hardware and software issues.	**HS-SAS-HW-03.** Apply systematic troubleshooting techniques to identify, diagnose, and resolve issues in computing systems.
3	**E3-SAS-HW-01.** Explain how the basic hardware and software components of a computing system work together to perform input/output (I/O) operations.		
4	**E4-SAS-HW-01.** Describe the use of sensors in your environment (e.g., motion sensor doors, classroom lights).		
5	**E5-SAS-HW-01.** Articulate steps to implement common solutions (e.g., turn off/on, plug in) for hardware and software issues (e.g., lack of sound, will not turn on), using accurate terminology.		

Networks

<table>
<tr><th colspan="2">Grades PreK–5</th><th>Grades 6–8</th><th>Grades 9–12 Foundations</th></tr>
<tr><td>PreK–K</td><td>—</td><td rowspan="6">MS-SAS-NW-04. Analyze the basic components, advantages, and disadvantages of wired and wireless local area networks (LANs).

MS-SAS-NW-05. Explain the key components of the Internet, their roles and functionality, and how they contribute to its resilience.

MS-SAS-NW-06. Model how information travels securely across digital networks through the use of physical and software tools.</td><td rowspan="6">HS-SAS-NW-04. Analyze how the Internet functions as a network of networks, including similarities and differences between the Internet and other types of networks in terms of structure, protocols, and scalability.

HS-SAS-NW-05. Diagram a computing system, including hardware, software, and a network, that can process information, accomplish a task, and/or solve a problem.

HS-SAS-NW-06. Analyze a network’s structure, its components, and optimization strategies.</td></tr>
<tr><td>1</td><td>—</td></tr>
<tr><td>2</td><td>—</td></tr>
<tr><td>3</td><td>E3-SAS-NW-02. Explain how computing devices connect to the Internet using wires (e.g., Ethernet) or wireless signals (e.g., Wi-Fi).</td></tr>
<tr><td>4</td><td>E4-SAS-NW-02. Illustrate how people access the global network of the Internet to access information and communicate with each other.</td></tr>
<tr><td>5</td><td>E5-SAS-NW-02. Model how information in a network is broken down into packets (smaller pieces), transmitted between devices, and reassembled.</td></tr>
</table>

Security

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-SAS-SC-02.** Apply practices for keeping personal data secure.	**MS-SAS-SC-07.** Classify common types of cyberattacks, including social engineering and malware. **MS-SAS-SC-08.** Evaluate the vulnerabilities of computing systems and data centers—the physical vulnerabilities (e.g., natural disasters) and intentional bad actors (e.g., data breaches, theft, dark patterns). **MS-SAS-SC-09.** Compare and contrast different types of cybersecurity and physical security measures and the trade-offs for users, data, and devices. **MS-SAS-SC-10.** Describe Defense in Depth strategies and what steps to take in response to cybersecurity incidents in simple networks.	**HS-SAS-SC-07.** Distinguish among the different types of cyberattacks that affect information security for individuals and organizations. **HS-SAS-SC-08.** Diagram computing systems that integrate security protocols, incorporating user-centered design principles such as user research, prototyping, and iterative design. **HS-SAS-SC-09.** Evaluate network vulnerabilities and mitigation strategies. **HS-SAS-SC-10.** Analyze the costs of cybersecurity breaches and social engineering attacks for individuals, industries, and governments.
1	**E1-SAS-SC-02.** Evaluate how sharing information online might reveal personally identifiable information (PII) and other details to people other than the intended recipients.		
2	**E2-SAS-SC-02.** Explain processes for maintaining secure personal and private information.		
3	**E3-SAS-SC-03.** Evaluate how authentication, different levels of access, and security measures help protect information on personal and public devices and networks.		
4	**E4-SAS-SC-03.** Model different methods of encryption and decryption for keeping information secure.		
5	**E5-SAS-SC-03.** Justify the importance of monitoring and updating security measures to prevent unauthorized access to information and other harms.		

Impacts of Computing Systems

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-SAS-IM-03.** Categorize computing tools based on what tasks they accomplish and how they might benefit the user.	**MS-SAS-IM-11.** Examine differences in access to computing systems, based on personal and social factors, including physical ability, geographic location, socioeconomic status, and age. **MS-SAS-IM-12.** Explain how widely used computing systems have helped society solve problems while simultaneously creating new ones (e.g., cyberbullying, environmental impacts).	**HS-SAS-IM-11.** Debate the trade-offs of global access to computing systems for society in terms of societal norms, interactions, and digital engagement. **HS-SAS-IM-12.** Evaluate the rationales behind laws, policies, and best practices governing the design and use of computing systems (e.g., user agreements, acceptable use policies, Children's Online Privacy Protection Act [COPPA], Digital Millennium Copyright Act Section 508). **HS-SAS-IM-13.** Investigate the societal and environmental impacts of computing systems and the physical infrastructure that supports them (e.g., data centers, servers, and mobile devices).
1	**E1-SAS-IM-03.** Describe the benefits and harms that arise from an individual's use of various computing tools.		
2	**E2-SAS-IM-03.** Evaluate an individual's role in responsibly using computing tools.		
3	**E3-SAS-IM-04.** Describe the benefits and harms of widely used computing platforms to an individual's life and human connections.		
4	**E4-SAS-IM-04.** Analyze the impacts of widely used computing systems and networks on ecosystems and the environment.		
5	**E5-SAS-IM-04.** Collaborate on an improved design for a computing system so it can be better used by people with different needs, abilities, and ways of thinking.		

Computing and Society

History of Computing

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-CAS-HC-01.** Identify computing technologies used in modern daily life that have changed significantly in the last 50 years.	**MS-CAS-HC-01.** Evaluate how challenges in different communities around the world have led to computing innovations. **MS-CAS-HC-02.** Compare the roles of individuals, communities, organizations, and governments in advancing computing technologies. **MS-CAS-HC-03.** Analyze intended and unintended impacts of historical computing technologies on society and the environment.	**HS-CAS-HC-01.** Differentiate major eras in computing history and key advancements by notable individuals and organizations. **HS-CAS-HC-02.** Evaluate policies and legislation designed to encourage ethical innovation and minimize societal risks associated with technology. **HS-CAS-HC-03.** Explain, with examples, how a historical computing practice led to a specific present-day ethical concern. **HS-CAS-HC-04.** Compare differing historical computing innovations and their effects on present-day societal practices and the environment.
1	**E1-CAS-HC-01.** Explore the contributions of people from different cultures, backgrounds, and time periods who helped shape computing technologies.		
2	**E2-CAS-HC-01.** Compare daily life before and after the implementation or adoption of a computing technology.		
3	**E3-CAS-HC-01.** Research how computer scientists and their work have shaped your local community.		
4	**E4-CAS-HC-01.** Describe how computing technologies evolve in response to societal needs and changes.		
5	**E5-CAS-HC-01.** Examine how a computing innovation changed the way people lived, worked, or communicated over time.		

Emerging Technologies

Grades PreK–5		Grades 6–8	Grades 9–12 Foundations
PreK–K	**EK-CAS-ET-02.** Describe how computing technology, used in daily life, at home, and at school, can help people.	**MS-CAS-ET-04.** Evaluate how design decisions in emerging technologies influence user experiences differently across different communities. **MS-CAS-ET-05.** Contrast the features, functionality, and characteristics of emerging technologies with technologies that came before. **MS-CAS-ET-06.** Describe ways that emerging technologies have influenced the development of new solutions or transformed existing processes across society and the environment. **MS-CAS-ET-07.** Explain how emerging technologies create new solutions that depend on expertise from multiple fields.	**HS-CAS-ET-05.** Explain how the computing principles underlying emerging technologies are being used in innovative ways. **HS-CAS-ET-06.** Describe how an emerging technology will impact an existing project or technology. **HS-CAS-ET-07.** Evaluate an emerging technology through multiple ethical perspectives. **HS-CAS-ET-08.** Identify how an emerging technology could lead to potential enhancements or alternative approaches for an existing computational solution. **HS-CAS-ET-09.** Evaluate the societal and environmental impacts of emerging technologies, including those that lead to inequities in access and outcomes.
1	**E1-CAS-ET-02.** Describe how technologies new to students create both benefits and harms in personal and family life.		
2	**E2-CAS-ET-02.** Hypothesize ways that new or improved technologies can solve a problem.		
3	**E3-CAS-ET-02.** Evaluate how people make choices about the use of emerging technologies, based on their needs and the consequences.		
4	**E4-CAS-ET-02.** Investigate intended and unintended consequences related to emerging technologies.		
5	**E5-CAS-ET-02.** Analyze the limitations of existing technologies and how emerging technologies change the way people work, behave, and communicate.		

Career Exploration

<table>
<tr><th colspan="2">Grades PreK–5</th><th>Grades 6–8</th><th>Grades 9–12 Foundations</th></tr>
<tr><td>PreK–K</td><td>EK-CAS-CE-03. Identify how people use digital devices in their daily work.</td><td rowspan="6">MS-CAS-CE-08. Analyze how professionals in different careers use computational thinking to solve real-world problems.

MS-CAS-CE-09. Examine how changes in technology can create new jobs or change how people work.

MS-CAS-CE-10. Investigate how professionals in computing careers address ethical dilemmas.

MS-CAS-CE-11. Illustrate how technology-driven challenges require integrating computing knowledge and skills with expertise from diverse fields.</td><td rowspan="6">HS-CAS-CE-10. Evaluate how computing knowledge and skills align with personal interests and career aspirations.

HS-CAS-CE-11. Investigate how professionals apply CS in their careers, drawing from their personal narratives.</td></tr>
<tr><td>1</td><td>E1-CAS-CE-03. Describe how computing is used in a variety of industries and careers.</td></tr>
<tr><td>2</td><td>E2-CAS-CE-03. Discuss ways personal interests connect to computing in different industries and careers.</td></tr>
<tr><td>3</td><td>E3-CAS-CE-03. Explain how computing technologies and skills are used across different industries (e.g., healthcare, transportation, and entertainment).</td></tr>
<tr><td>4</td><td>E4-CAS-CE-03. Examine how professionals across industries adopt new computing technologies and develop new expertise throughout their careers.</td></tr>
<tr><td>5</td><td>E5-CAS-CE-03. Investigate how professionals collaborate with computing technologies to solve problems creatively, accurately, and efficiently.</td></tr>
</table>

Specialty Standards for High School

Naming Conventions for Specialty Standards

Each of the identifiers for specialty standards follows a naming convention similar to the one for foundational standards:

Level		Focus area		Sub-area		Number
XX	-	YYY	-	ZZ	-	##

The first two characters indicate Specialty I (S1) or Specialty II (S2) standards. The three characters in the next section indicate the focus area:

Abb	Focus Area
SWD	Software Development
CYB	Cybersecurity
AIN	Artificial Intelligence
PHY	Physical Computing
DSC	Data Science
GMD	Game and Interactive Media Design
XCS	X + CS

The next two characters indicate the sub-area. Specialty I and Specialty II standards within a focus area do not necessarily have identical sub-areas, so they are not listed here. The last two digits are the standard number. Numbering begins with 01 for Specialty I standards in each focus area. Specialty II standards continue the numbering from Specialty I standards and do not restart from 01.

Software Development

Specialty I	Specialty II
Program Development **S1-SWD-PD-01.** Construct programs to model real-world systems, considering factors such as scale, complexity, and the need for simplification. **S1-SWD-PD-02.** Demonstrate an understanding of user experience (UX) principles and software engineering best practices. **Data Handling** **S1-SWD-DH-03.** Apply different fundamental data structures (e.g., arrays, linked lists, stacks, queues, trees, graphs) to solve various computational problems. **Testing and Refining Code** **S1-SWD-TR-04.** Design test cases that effectively exercise the functionality of a program, considering potential edge cases, error conditions, and user inputs to ensure thorough and systematic testing. **S1-SWD-TR-05.** Employ systematic debugging techniques to identify, isolate, and fix program errors, utilizing debugging tools and effective problem-solving strategies.	**Program Development** **S2-SWD-PD-06.** Apply an industry-standard software development process to plan software projects effectively. **S2-SWD-PD-07.** Use the features of an integrated development environment (IDE) to enhance the software development process, including code editing, debugging, version control, and project management. **S2-SWD-PD-08.** Develop software in collaboration with others, using version control tools to manage contributions. **S2-SWD-PD-09.** Modify existing algorithms to improve their efficiency, considering factors such as data structures and algorithmic paradigms (e.g., divide-and-conquer). **S2-SWD-PD-10.** Apply an industry-standard software development process to deliver software projects effectively.

Cybersecurity

Specialty I	Specialty II
Foundational Cybersecurity Skills **S1-CYB-FC-01.** Explain the concepts of IP and MAC addresses, their roles in network communication, and the potential security implications of misconfigured or exposed addresses. **S1-CYB-FC-02.** Understand how networking hardware, including servers, switches, routers, endpoints, and firewalls, can be used to make a network more secure. **S1-CYB-FC-03.** Classify a network by noting its protocols, topologies, and addressing. **S1-CYB-FC-04.** Analyze the implementation of major networking protocols (e.g., TCP/IP, HTTP, HTTPS) to understand their underlying mechanisms and identify potential vulnerabilities. **S1-CYB-FC-05.** Analyze the security implications of different network topologies (e.g., bus, star, mesh) and identify potential vulnerabilities and mitigation strategies. **S1-CYB-FC-06.** Systematically apply diagnostic tools and techniques to resolve network connectivity issues, considering factors such as network topology, protocols, and security configurations.	**Foundational Cybersecurity Skills** **S2-CYB-FC-07.** Use command-line programming to audit system processes, monitor network traffic, and scan for vulnerabilities. **S2-CYB-FC-08.** Use scripting languages (e.g., Python, PowerShell) to automate tasks, analyze data, and perform security operations, understanding the potential security risks associated with improper scripting practices. **S2-CYB-FC-09.** Use programming languages (e.g., C++, Python) to interface with hardware components, such as microcontrollers and sensors, to implement security measures. **Network Threats, Mitigation, and Incident Response** **S2-CYB-NT-10.** Classify a security threat using the CIA triad (confidentiality, integrity, availability), states of data (at rest, in transit, in use), and types of control (preventative, detective, corrective). **S2-CYB-NT-11.** Identify the source of a threat using levels from models (e.g., the Open Systems Interconnection [OSI] model and the TCP/IP model) and accurate terminology for network parameters (e.g., protocols, ports, and IP addresses). **S2-CYB-NT-12.** Evaluate the security risks associated with small office/home office (SOHO)/home networks, including vulnerabilities in routers, wireless networks, and Internet-connected devices. **S2-CYB-NT-13.** Implement best practices to mitigate security risks associated with SOHO/home networks, including vulnerabilities in routers, wireless networks, and Internet-connected devices.

Specialty I	Specialty II
	Network Threats, Mitigation, and Incident Response, continued **S2-CYB-NT-14.** Describe the vulnerabilities and corresponding threats they pose to individuals and organizations, as well as potential remediation strategies. **S2-CYB-NT-15.** Implement security measures, such as using virtual private networks (VPNs) and secure browsing practices, to protect sensitive information. **S2-CYB-NT-16.** Implement best practices to mitigate common application security vulnerabilities, including input sanitization and validation, output encoding, and secure session management. **S2-CYB-NT-17.** Implement database access controls, including user authentication, authorization, and data encryption, to protect sensitive information from unauthorized access and data breaches. **S2-CYB-NT-18.** Evaluate security risks, assess their potential impacts, and implement appropriate risk mitigation strategies, such as vulnerability assessments, penetration testing, and security awareness training. **S2-CYB-NT-19.** Analyze business continuity and disaster recovery plans, identify critical systems and processes, and develop strategies to minimize downtime and maintain business operations in the event of a security incident or natural disaster. **S2-CYB-NT-20.** Discuss incident response plans, identify key steps in the incident response process, and simulate real-world incident response scenarios to develop critical thinking and problem-solving skills. **S2-CYB-NT-21.** Evaluate the security implications of emerging technologies, such as AI, blockchain, and the Internet of Things (IoT), and identify potential vulnerabilities and mitigation strategies.

Specialty I	Specialty II
	Ethical Considerations, Policy, and Compliance **S2-CYB-EC-22.** Analyze the potential benefits, risks, and ethical implications of AI in cybersecurity, including its use in threat detection, incident response, and offensive cyber operations. **S2-CYB-EC-23.** Explain the importance of cybersecurity policies in protecting organizational assets and mitigating risks. **S2-CYB-EC-24.** Identify key components of effective security policies, such as access control, incident response, and data protection. **S2-CYB-EC-25.** Apply ethical hacking techniques and tools to identify and exploit vulnerabilities in systems and networks, with the goal of improving security and protecting sensitive information. **S2-CYB-EC-26.** Analyze the impact of regulations (e.g., GDPR, HIPAA, PCI DSS) on organizational security policies and procedures, and identify compliance requirements. **Career Exploration** **S2-CYB-CE-27.** Research cybersecurity career paths, including cybersecurity analyst, penetration tester, digital forensics analyst, and information security engineer. **S2-CYB-CE-28.** Identify the necessary skills and certifications for cybersecurity career paths.

Artificial Intelligence

Specialty I	Specialty II
AI Concepts, Development, and Application **S1-AIN-CD-01.** Apply data collection and cleaning techniques to prepare data for AI analysis and modeling. **S1-AIN-CD-02.** Apply foundational knowledge and skills in prompt engineering to generate outputs from AI models. **S1-AIN-CD-03.** Explain the principles of natural language processing (NLP) and their application in AI. **S1-AIN-CD-04.** Describe how representations and reasoning are implemented in AI, including feature extraction, vector space models, and the application of ML algorithms. **S1-AIN-CD-05.** Develop a computational artifact that integrates an existing AI tool. **AI History, Ethics, and Human Interaction** **S1-AIN-HE-06.** Evaluate the role of human decision-making in the development of emerging technologies. **S1-AIN-HE-07.** Summarize the historical development, current capabilities, potential future applications, and ethical implications of AI. **S1-AIN-HE-08.** Analyze machine learning model optimization techniques, accuracy metrics, decision-making processes, and ethical considerations related to model use.	**AI Concepts, Development, and Application** **S2-AIN-CD-09.** Compare different computational solutions to real-world problems, including how AI tools and libraries are used in different solutions. **S2-AIN-CD-10.** Interpret sensory data, using machine perception concepts such as feature extraction, object recognition, resolving ambiguity, and determining context. **S2-AIN-CD-11.** Apply data analysis techniques, including regression models and probabilistic reasoning, to extract insights from datasets and make informed predictions. **S2-AIN-CD-12.** Analyze the application of computer vision, image and video analysis, and object detection in real-world scenarios, using sensor data and AI models. **S2-AIN-CD-13.** Develop intelligent robotic systems that interact with the real world through electronics, mechanics, cameras, sensors, and AI models. **AI History, Ethics, and Human Interaction** **S2-AIN-HE-14.** Conduct user interviews to gain insights into AI user needs and experiences. **S2-AIN-HE-15.** Analyze the ethical implications of AI, including bias, fairness, transparency, and accountability. **S2-AIN-HE-16.** Critically analyze the potential biases and limitations of advanced ML techniques. **S2-AIN-HE-17.** Apply bias mitigation strategies to ensure fairness in student-created AI systems.

Physical Computing

Specialty I	Specialty II
System Development and Security	**System Development and Security**
S1-PHY-SD-01. Use a physical computing device to accomplish a real-world task or solve a real-world problem, demonstrating an understanding of the device's capabilities and limitations.	**S2-PHY-SD-09.** Employ the Internet of Things (IoT) devices to collect and transmit real-world data, enabling remote monitoring and control of physical systems.
S1-PHY-SD-02. Analyze the technical specifications and limitations of various physical computing devices (e.g., sensors, actuators, microcontrollers) to inform design decisions for physical computing projects.	**S2-PHY-SD-10.** Implement networks to connect physical devices, enabling communication, data sharing, and remote control, considering factors such as network protocols, security, and scalability.
S1-PHY-SD-03. Implement basic electrical circuits, including power sources, resistors, and other components, to power and control physical computing devices.	**S2-PHY-SD-11.** Develop applications, such as mobile apps, virtual reality experiences, and AI agents, to interact with and control physical devices, extending their functionality and user engagement.
S1-PHY-SD-04. Integrate sensors and peripherals with physical computing devices to extend their functionality and gather real-world data for analysis and control.	**S2-PHY-SD-12.** Apply software development methodologies, such as Agile or Scrum, to manage the iterative development process of physical computing projects, including planning, design, implementation, testing, deployment, and evaluation.
S1-PHY-SD-05. Develop circuits involving motors and microcontrollers to create electromechanical systems, considering factors such as power requirements, motor types, and control algorithms.	**Collaboration and Communication**
S1-PHY-SD-06. Implement code to control physical devices, leveraging programming languages and environments tailored for physical computing, and remotely debugging issues that arise in both hardware and software.	**S2-PHY-CC-13.** Effectively communicate physical computing solutions to diverse audiences, explaining the underlying concepts, design decisions, and potential applications, and provide clear instructions for replication.
S1-PHY-SD-07. Apply the engineering design process (identify a need or problem, research, design, prototype, test, evaluate, and iterate) to develop and refine physical computing solutions, including debugging and troubleshooting hardware and software components.	**S2-PHY-CC-14.** Collaborate with others to effectively design, develop, and test physical computing projects, sharing responsibilities, resolving conflicts, and communicating progress and outcomes.
S1-PHY-SD-08. Evaluate the security implications of physical computing projects, including data privacy, unauthorized access, and potential vulnerabilities, and implement measures to mitigate risks.	**S2-PHY-CC-15.** Use collaborative source control systems (e.g., Git) to manage project versions, track changes, and facilitate teamwork in physical computing projects.
	Career Exploration
	S2-PHY-CE-16. Explore career paths in physical computing and related fields, understanding the skills, knowledge, and creativity required to design, develop, and implement innovative solutions.

Data Science

Specialty I	Specialty II
Data Management **S1-DSC-DM-01.** Combine datasets to facilitate data analysis and modeling. **S1-DSC-DM-02.** Write code to filter, group, summarize, and transform data in preparation for data visualization and data analysis. **S1-DSC-DM-03.** Document data manipulations to facilitate replication of data cleaning and analysis. **Data Analysis and Visualization** **S1-DSC-AV-04.** Evaluate the impact of missing data within datasets, using computational methods. **S1-DSC-AV-05.** Implement effective techniques to extract information from both structured and unstructured data sources. **S1-DSC-AV-06.** Analyze data science problems, using statistical approaches. **S1-DSC-AV-07.** Demonstrate how common graphical conventions are used in data visualizations and how breaking these conventions can lead to misleading interpretations. **S1-DSC-AV-08.** Ensure accurate data documentation, including source documentation, data dictionary, data lineage, and version control.	**Advanced Data Analysis and Predictive Modeling** **S2-DSC-AP-15.** Analyze large-scale datasets, using data analysis platforms and distributed computing frameworks. **S2-DSC-AP-16.** Evaluate the practical implications of data analysis results. **S2-DSC-AP-17.** Develop predictive models to make accurate forecasts. **Domain-Specific Applications** **S2-DSC-DS-18.** Analyze data from devices that humans interact with (e.g., cameras, sensors, wearables) to inform decision-making. **S2-DSC-DS-19.** Investigate the intersection of data science with other fields, such as biology, finance, and social sciences, to address real-world problems. **Ethical and Legal Considerations** **S2-DSC-EL-20.** Assess potential biases in data and models. **S2-DSC-EL-21.** Analyze data science's ethical, legal, and societal implications, including data privacy, security, bias, missing data, and responsible use. **S2-DSC-EL-22.** Demonstrate ethical and responsible data practices through real-world applications, including analyzing case studies and creating awareness campaigns. **S2-DSC-EL-23.** Critically evaluate new approaches to data collection, usage, and governance.

Specialty I	Specialty II
Predictive Modeling and Evaluation S1-DSC-PM-09. Explain how training data influences predictions and recognize the limitations of ML models. S1-DSC-PM-10. Assess the appropriateness of predictive models for the specific problem being addressed. S1-DSC-PM-11. Build predictive models, using a variety of machine learning techniques. **Tool Proficiency** S1-DSC-TP-12. Analyze complex datasets, using computational tools and libraries. S1-DSC-TP-13. Develop data science projects in integrated development environments (IDEs). **Career Exploration** S1-DSC-CI-14. Investigate the skills and knowledge needed to pursue a career in data science.	

Game and Interactive Media Design

Specialty I	Specialty II
Program Development **S1-GMD-PD-01.** Create user-friendly interfaces (UIs) for interactive media experiences, considering factors such as accessibility and visual appeal. **S1-GMD-PD-02.** Implement game logic to create dynamic gameplay experiences, including character movement, object interactions, environmental interactions, and player feedback. **S1-GMD-PD-03.** Create storyboards to communicate game narratives, mechanics, and user experience, emphasizing visual storytelling and pacing. **Testing and Refining Code** **S1-GMD-TR-04.** Develop interactive experiences that use principles of user-centered design, prototyping, and iterative development to create engaging interactive media. **S1-GMD-TR-05.** Design interactive simulations that model real-world systems or phenomena by applying systems thinking, utilizing data modeling techniques, and incorporating user experience principles to create engaging and accurate representations.	**Program Development** **S2-GMD-PD-06.** Design game interactions that utilize a variety of input devices (e.g., joysticks, gamepads, motion controllers, VR headsets, touchscreens) to enhance gameplay, immersion, and player experience. **S2-GMD-PD-07.** Apply principles of universal design to create games and interactive media that are inclusive and accessible for all users. **S2-GMD-PD-08.** Apply object-oriented programming (OOP) principles and techniques to design, develop, and maintain game systems, resulting in modular, reusable, and maintainable code. **S2-GMD-PD-09.** Collaborate effectively within a team environment to plan, develop, and iterate on a game project, demonstrating strong communication, interpersonal, and problem-solving skills, and contributing meaningfully to the success of the team. **S2-GMD-PD-10.** Develop fundamental 2D and 3D animations for game assets, demonstrating proficiency in keyframing and basic animation principles. **Architecture** **S2-GMD-AR-11.** Explain the role of graphics processing units (GPUs) in game development, including their impacts on rendering performance, visual fidelity, and the overall gaming experience. **S2-GMD-AR-12.** Understand the key architectural features of modern GPUs and their implications for game development. **Testing and Refining Code** **S2-GMD-TR-13.** Apply human-centered design principles to the design and development of engaging and meaningful game experiences, considering player motivation, emotion, and behavior. **S2-GMD-TR-14.** Conduct basic A/B tests comparing two versions of a game element (e.g., a feature, mechanic, or UI element) to determine which version performs better based on specific metrics (e.g., player engagement, retention, conversion rates).

X + CS

Specialty I	Specialty II
S1-XCS-XC-01. Identify and explain connections between CS concepts and practices (e.g., data, abstraction) and concepts and practices in a non-CS discipline (X). **S1-XCS-XC-02.** Apply computational thinking to reinterpret problems and design solutions within a non-CS discipline (X), demonstrating how computing concepts can enhance understanding and innovation across fields. **S1-XCS-XC-03.** Illustrate how advancements in CS and in a non-CS discipline (X) have mutually influenced each other.	**S2-XCS-XC-04.** Integrate concepts and practices from CS with concepts and practices from a non-CS discipline (X) to solve a problem and investigate key ideas, methods, or processes within X. **S2-XCS-XC-05.** Model data within a non-CS discipline (X), using data visualizations and computational methods. **S2-XCS-XC-06.** Apply insights from a non-CS discipline (X) to solve computational problems. **S2-XCS-XC-07.** Assess how well algorithms solve problems within a non-CS discipline (X) by analyzing their accuracy, efficiency, and relevance to the intended goal.

References

Association for Computing Machinery (ACM). (2018). *ACM code of ethics and professional conduct.* https://www.acm.org/code-of-ethics

Interaction Design Foundation (IDF). (n.d.a). *Human-centered design (HCD).* https://www.interaction-design.org/literature/topics/human-centered-design

Interaction Design Foundation (IDF). (n.d.b). *Humanity-centered design (HCD).* https://www.interaction-design.org/literature/topics/humanity-centered-design

International Technology and Engineering Educators Association (ITEEA). (2020). *Standards for technological and engineering literacy: The role of technology and engineering in STEM education.* https://www.iteea.org/stel

K–12 Computer Science Framework. (2016). http://www.k12cs.org

Learning for Justice. (n.d.). *Social justice standards: The Learning for Justice anti-bias framework.* https://www.learningforjustice.org/frameworks/social-justice-standards

National Council for the Social Studies (NCSS). (2010). *National curriculum standards for social studies: A framework for teaching, learning, and assessment.*

National Governors Association Center for Best Practices (NGA Center) & Council of Chief State School Officers (CCSSO). (2010). *Common Core State Standards for Mathematics.* https://corestandards.org/wp-content/uploads/2023/09/Math_Standards1.pdf

National Institute of Standards and Technology (NIST). (2021). *Human centered design (HCD).* https://www.nist.gov/itl/iad/visualization-and-usability-group/human-factors-human-centered-design

Partnership for 21st Century Skills. (2009). *P21 framework definitions.* https://files.eric.ed.gov/fulltext/ED519462.pdf

Vinney, C. (2023). *What is human-centered design? Everything you need to know.* UX Design Institute. https://www.uxdesigninstitute.com/blog/what-is-human-centered-design/

