New Hampshire Department of Education

Serving New Hampshire's Education Community

New Hampshire Computer Science Standards

Rev. 2018

Key Supporting NH Organizations

CS4NH

CSTA-NH

NH High Tech Council — Advancing Innovation

NHSTE — www.nhste.org

Key Supporting National Organizations

K12 CS

CSTA

Part 1: Context and Considerations

Acknowledgements

The 2018 NH K-12 Computer Science Standards is the result of the hard work of numerous individuals across the country and state. Work in NH has been informed and supported by many national-level associations, alliances and non-profit organizations, including the Computer Science Teachers Association (CSTA), Expanding Computing Education Pathways (ECEP) Alliance, CSforAll Coalition, and Code.org.

Many important stakeholders, including numerous educators in NH, have participated in national-level leadership to advance K-12 Computer Science education. The development of the Standards, and the associated work of providing support to organizations who play a role in implementing the Standards, has been and will continue to be carried out by countless stakeholders across the state.

We would like to recognize the following organizations, who are providing leadership in advancing K-12 Computer Science in NH: The CS4NH Alliance, NH High Technology Council (NHHTC), NH Charitable Foundation (NHCF), University System of NH (USNH), Community College System of NH (CCSNH), NH Society for Technology in Education (NHSTE), and NH Computer Science Teachers Association (NH-CSTA).

To see individuals in the NH Computer Science Standards revision team, and CS4NH Alliance steering committee members, who contributed significantly to the development and review of these standards, please see the appendices.

Vision: Computer Science for All Students in New Hampshire

> “Computer science and the technologies it enables now lie at the heart of our economy, our daily lives, and scientific enterprise. [...] To be a well-educated citizen as we move toward an ever-more computing-intensive world and to be prepared for the jobs of the 21st Century, students must have a deeper understanding of the fundamentals of computer science.” [ACM]

In order to be an informed, engaged, and productive citizen in our State and our Nation, it is imperative that students learn the fundamental skills and knowledge of computer science. Computer science and computing technologies affect us socially, politically, and economically.

- Computer science is changing how we interact with our environment and with one another.
- Computer science is changing how we interact with our political leaders and institutions.
- Computer science is disrupting every industry, creating new industries, and driving new scientific and engineering breakthroughs.

The NH K-12 Computer Science Standards will guide educators as they seek to respond to these changes. They will specify clear learning objectives for students and will serve as a resource for local development and/or adoption of curriculum, instructional materials, and performance assessments.

The standards will help empower educators and students, in order to:

- “critically engage in public discussion on computer science topics;
- “develop as learners, users, and creators of computer science knowledge and artifacts;
- “better understand the role of computing in the world around them; and
- “learn, perform, and express themselves in other subjects and interests.” [K12CS]

About the Standards

Overview of Standards Development

Committee Formation: In August 2017, a rationale and plan was presented to the NH State Board of Education for the development of academic standards for Computer Science. An application for membership was widely distributed and a voluntary committee was formed. This committee was composed of various educator stakeholders, including: primary educators, secondary educators, K-12 administrators, and higher education faculty.

Background Research: After conducting background research, the committee unanimously elected to build our standards using the recently released K-12 Computer Science Framework and the Computer Science Teachers Association (CSTA) K-12 Computer Science Standards as primary sources. The initial development and review of these national-level documents involved many important stakeholders, including several in NH.

Draft One development: The committee determined the structure of the Standards and established subcommittees: Editorial, Primary Education, Secondary Education. This first draft of the standards, which included only organizational structure and primary source materials, was released for public review in October 2017.

Draft Two development: The subcommittees performed detailed reviews of source documents and other references, including a standard-by-standard review of the CSTA K-12 Computer Science Standards, and produced original content (Introduction, Background, Implementation Guidance, and Appendices) and recommendations to the full committee. The committee determined that the CSTA Standards are appropriate for NH's purposes and recommended to adopt them with minimal modification.

Draft Two public input: Draft 2 will be released publicly in May 2018. It will be distributed electronically with a feedback survey, and presented in a statewide listening tour. It will be reviewed by the CS4NH Advisory committee, the Pre-Engineering and Technology Advisory Council (PETAC), and by members of the NH State Board of Education. It will be reviewed by boards of several of NH's professional educator associations.

State Board approval: Barring any unforeseen setbacks, approval for NH's K-12 Computer Science Standards will be requested in Summer 2018.

Objectives for Standards

The following objectives were central to the standards development, and should also be considered when implementing the standards. Adapted from [K12CS].

- **Objective 1: Rigor**
  - Establish and articulate the appropriate level of rigor in computer science to prepare all students for success in college and careers.
- **Objective 2: Focus / Manageability**
  - Prioritize the concepts and skills that should be acquired by students. A sharpened focus helps ensure that the knowledge and skills students are expected to learn are important and manageable in any given grade or course.
- **Objective 3: Specificity / Clarity**
  - Specify what is computer science, and distinguish between computer science and other uses of computers in a K-12 setting.
  - Provide sufficient detail to convey the level of performance expected without being overly prescriptive.
- **Objective 4: Equity / Diversity / Accessibility**
  - Allow for engagement by all students and allow for flexibility in how students may demonstrate proficiency. The standards are based on the belief that all students, regardless of race, gender, socioeconomic class, or disability, when given appropriate support, can learn all of the concepts and practices described herein.
- **Objective 5: Coherence / Progression**
  - Organized as progressions that support student learning of content and practices over multiple grades.
  - Convey a unified vision of the discipline, establishing connections among the major areas of study and showing a meaningful progression of content across grade levels and grade spans.
- **Objective 6: Measurability**
  - Objective and measurable. Focus on the results, rather than the processes of teaching and learning.
- **Objective 7: Integration of Practices and Concepts**
  - Integrate the computer science practices with the concept statements. Students learn by doing.
- **Objective 8: Connections to Other Disciplines**
  - Make intentional connections between computer science and other disciplines, so that students can understand how computer science affects their world.
  - Promote more coherent education experiences for students.

Background Information

Computer Science Overview

“As the foundation for all computing, computer science is defined as “the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society” [Tucker et. al., cited in K12CS]

Computer Science (CS) includes five core concept areas: ● computing systems, ● networks and the Internet, ● data and analysis, ● algorithms and programming, and ● impacts of computing.	In addition, ***Computational Thinking (CT)***, includes core practices of: ● recognizing and defining computational problems ● developing and using abstractions ● creating computational artifacts ● testing and refining computational artifacts

Computer Science in Context

Information and communication technologies (ICT) in schools

Computer science should not be conflated with other aspects and uses of computer technology in schools, including:

- ***“Computer literacy [i.e. Digital Literacy, ICT Literacy]*** refers to the general use of computers and programs, such as productivity software. Previously mentioned examples include performing an Internet search and creating a digital presentation.
- ***“Educational technology*** applies computer literacy to school subjects. For example, students in an English class can use a web-based application to collaboratively create, edit, and store an essay online.
- ***“Digital citizenship*** refers to the appropriate and responsible use of technology, such as choosing an appropriate password and keeping it secure.
- ***“Information technology*** often overlaps with computer science but is mainly focused on industrial applications of computer science, such as installing and operating software rather than creating it. Information technology professionals often have a background in computer science.” [k12cs]

CS and STEM

> "Because CS is an active and applied field of Science, Technology, Engineering and Math (STEM) learning that allows students to engage in hands-on, real-world interaction with key math, science, and engineering principles, it gives students opportunities to be creators - not just consumers - in the digital economy..." [CSforALL]

CS and Math
Computer science and computation are fundamentally mathematical. Computing is built on mathematical principles including formal logic. Computing can be used to perform arithmetic and logical operations. Combining these operations allow computing to be used in the diverse ways we see today.

CS and Science
Science includes the systematic study of the structure and behavior of the physical and natural world through observation and experiment, and a systematically organized body of knowledge on a particular subject. Computer science includes the study of computation and algorithmic processes, which don't necessarily need to be implemented in machinery. For example, physical, chemical, and biological processes can all be explored in terms of computation, without necessarily involving any human-built computing devices. Computer science is a systematic study, and computer scientists have compiled a vast body of knowledge in this area.

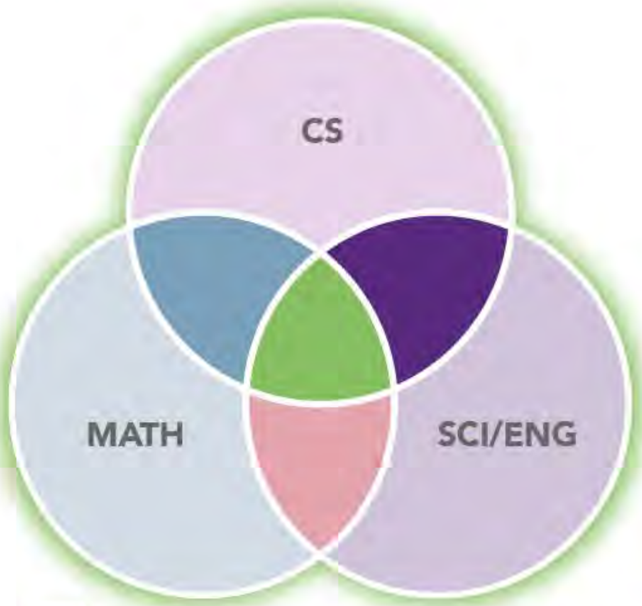
CS and Computer Modeling and Simulation
The term computational science refers to the use of computational tools and methods in science and engineering, such as modeling and simulation. Computer science informs the development of these tools. In practice, computational sciences involve both computer scientists and specialists in the other areas working together.

CS and Engineering
Engineering is concerned with the analysis, design, implementation, and use of engines, machines, structures, processes, etc. Engineered structures and processes can be physical (e.g., mechanical, chemical, biological, etc.), but they can also be virtual (e.g., computer software).

Computer scientists work out computer hardware, software, and network designs. Engineers turn those designs into working devices and systems. Engineers also often encounter unexpected results, which can then be taken into consideration by the scientists to inform developing theories.

See below for relationships between CS, Science, Engineering, and Math practices. (Image source [K12CS]).

RELATIONSHIPS BETWEEN COMPUTER SCIENCE, SCIENCE AND ENGINEERING, AND MATH PRACTICES



CS + Math

- **Develop and use abstractions**
  M2. Reason abstractly and quantitatively
  M7. Look for and make use of structure
  M8. Look for and express regularity in repeated reasoning
  CS4. Developing and Using Abstractions
- **Use tools when collaborating**
  M5. Use appropriate tools strategically
  CS2. Collaborating Around Computing
- **Communicate precisely**
  M6. Attend to precision
  CS7. Communicating About Computing

CS + Sci/Eng

- **Communicate with data**
  S4. Analyze and interpret data
  CS7. Communicating About Computing
- **Create artifacts**
  S3. Plan and carry out investigations
  S6. Construct explanations and design solutions
  CS4. Developing and Using Abstractions
  CS5. Creating Computational Artifacts
  CS6. Testing and Refining Computational Artifacts

CS + Math + Sci/Eng

- **Model**
  S2. Develop and use models
  M4. Model with mathematics
  CS4. Developing and Using Abstractions
  CS6. Testing and Refining Computational Artifacts
- **Use computational thinking**
  S5. Use mathematics and computational thinking
  CS3. Recognizing and Defining Computational Problems
  CS4. Developing and Using Abstractions
  CS5. Creating Computational Artifacts
- **Define problems**
  S1. Ask questions and define problems
  M1. Make sense of problems and persevere in solving them
  CS3. Recognizing and Defining Computational Problems
- **Communicate rationale**
  S7. Engage in argument from evidence
  S8. Obtain, evaluate, and communicate information
  M3. Construct viable arguments and critique the reasoning of others
  CS7. Communicating About Computing

* Computer science practices also overlap with practices in other domains, including English language arts. For example, *CS1. Fostering an Inclusive Computing Culture* and *CS2. Collaborating Around Computing* overlap with *E7. Come to understand other perspectives and cultures through reading, listening, and collaborations.*

CS and Career & Technical Education (CTE)

Computer science is sometimes also confused with Career & Technical Education (CTE) clusters and pathways. The ***Information Technology cluster in CTE*** includes the following pathways:

- Network Systems Pathway
- Information Support & Services Pathway
- Web & Digital Communications Pathway
- Programming & Software Development Pathway

While the above are examples of CS pathways, the K-12 CS concepts and practices expressed in this Standards are foundational skills and knowledge that are important for all students in order to be informed and productive citizens in the 21st century. They are applicable not just in the information technology occupations / pathways above, but also virtually every other cluster. Here are a few notable examples:

- ***Engineering / Manufacturing.*** Advanced manufacturing is distinguished by the use of technology such as robotics and automation, digital modeling and simulation, etc.
- ***Health science.*** One-third of all practitioners / technical occupations in healthcare are technical (technologists and technicians).
- ***Business management and finance.*** These sectors are being transformed by technology, notably the use of analytics - the systematic computational analysis of data or statistics.

Students who are interested in focusing on the design and development of computer hardware and software systems and networks are encouraged to consider CTE programs in Engineering and/or Information and Communication Technologies. Students who are interested in applying these technologies in other areas might consider other programs.

Program Implementation

Key Considerations

Leadership and Administration

Establish a STEM / Computer Science advisory board. This group could include teachers, administrators, school board members, parents, members of the business / industry community, and other community members. A well-composed group will be in a good position to make recommendations regarding STEM / Computer Science curriculum planning and implementation.

Establish implementation team(s). This is a group of educators within a school who will do the implementation work necessary to establish and strengthen robust K-12 computer science programs. A teacher or administrator can't go it alone - it requires vertical integration.

Educator certification. At the primary level, schools should seek to employ a digital learning specialist (i.e. technology integrator) and/or a computer science educator. At the secondary level, schools should seek to employ at least one certified Computer Science teacher.

Course classification. Courses that are clearly computer science, as specified in these standards, should be classified as such in Educator Information Systems (EIS). Computer science should be recognized as a content area and reflected in your department names.

Curriculum and Professional Learning

Seek out standards-aligned resources and take advantage of professional development. Teachers in the early grades are tasked with helping their students develop in a great breadth of disciplines and may be unfamiliar or uncomfortable with CS and/or technology in general. In secondary grades, schools might want to “convert” an educator from one subject area to CS. Utilizing comprehensive curriculum aligned with the CS standards and participating in professional development can help address these challenges.
(See also Appendix: Teaching & Learning Resources)

Incorporate and/or integrate CS into your current schedule and curricula. While you may want to make schedule changes to accommodate expanded CS offerings, this may not be necessary. We distinguish incorporation and integration as follows:

- **Incorporate** - add or strengthen dedicated CS content in time that is already a part of the schedule, such as Technology / Engineering education, or Library / Media education.
- **Integrate** - integrate CS content into the teaching and learning of content in other related areas, especially in STEM, but also in the Arts, Humanities and other areas.

Instruction and Assessment

Use inquiry and make meaningful experiences with your students. Give students time and space to pursue CS projects that are related to their personal interests, including unstructured learning time. Help students learn how to learn. Connect CS content to social and cultural contexts and what's happening in their community. These practices can help students be empowered in their learning, and develop critical-thinking and creativity.

Use project-based and problem-based learning. CS naturally lends itself to project-based and problem-based learning. Project-based learning allows students to take time to develop and refine a product. Problem-based learning begins by identifying a specific problem to solve and designing and implementing solutions. Recognize the importance of both process and product.

Help students develop communication and collaboration skills. Students should collaborate via group projects. Communication and collaboration skills should be explicitly developed - don't expect them to "just know." Students should develop technical communication and presentation skills. Encourage students to reflect upon what they've learned and created.

Consider lessons and resources that do not require computers. Many fundamental CS concepts can be explored without even using a computer (e.g. CS Unplugged). Even in programming, algorithms can and should be worked out on paper or whiteboards using flowcharts and pseudocode. Don't be too dependent on online resources and tools - have a backup plan in case there are network problems.

Use a variety of assessment methods. Use continuous formative assessment to gather data that you can use to adapt and personalize your student's learning experience. Let students demonstrate their knowledge in a variety of ways to they can show their strengths. For example, portfolios, presentations, connect with inquiry / PBL (above). Encourage peer-to-peer feedback. Stress that CS is an iterative process and they can learn from their mistakes to improve their work. When CS is integrated in the school curriculum, students can use CS to demonstrate their knowledge in other subject areas (e.g. by creating an app).

Extended Learning Opportunities

Provide unstructured time for students to explore CS. Giving students unstructured time with programming tools will promote creativity and establish confidence in applying their CS knowledge. Facilitators do not need to be CS experts, but they should be aware of resources and tools that students can use to learn.

Sign up for expos and competitions. These programs can greatly help students develop presentation, collaboration skills, and more. They also allow students to interact with peers at other schools.

Work with your community. Encourage your students to get entrepreneurial. They can find clients in the community and work with them to design and develop a solution to a real problem. Find community members who want to give back and give them an opportunity to work with your students.

Program Progressions / Pathways

A strong K-12 CS district or school offers students time dedicated specifically to CS education, organized in a coherent progression, and also integrates CS with other areas.
(See also Appendix: Teaching & Learning Resources)

Primary (approx. K-6)

As we prepare our youngest learners for the demands of tomorrow, we need to acknowledge the role that computer science can play in their acquisition of 21st century skills. When elementary students are required to engage in computational thinking and solve real-world problems using technology, they are honing their ability to think critically, be creative, collaborate, and communicate. These skills, along with basic technological and digital literacy, are increasingly desired by their future teachers and employers.

Teaching computer science at the elementary level provides a solid foundation for students to build upon prior knowledge and experience real-world application of technology skills. However, establishing a pathway for achieving computer science standards can be a challenge due to barriers like scheduling and staffing. In contrast to middle and high schools, where computer science can be taught as a standalone course, elementary schools are more likely to require a flexible approach when it comes to implementing a computer science curriculum in grades K through 5.

Incorporate and/or integrate CS.
- Scenario A: Incorporate CS units as part of a library or technology “special” or “related arts” class.
- Scenario B: Integrate CS to design lessons that support both content area curriculum and CS concepts.

Utilize play-based learning in early childhood education
Children develop social and emotional skills through playful interactions with peers and adults, and research continually shows these interactions can have significant impacts on children’s learning and development. These strong affective, behavioral, and cognitive competencies provide the foundation for successful learning and development.

"Powerful Ideas" in Early Childhood CS Education

1. **Social and emotional learning.**
2. **Patterns**
3. **Problem-solving**
4. **Representation**
5. **Sequencing**

For more information, please visit "Computer Science in Early Childhood Education." [K12CS]

Secondary (approx. 7-12)

First secondary-level courses in CS should take a big-picture view of computer science, addressing each of the core content areas and practices. As students advance, they should have opportunities to explore in depth more specific areas of CS.

The recommendations below are intended as a menu of options that schools can explore and implement in an appropriate timeline. They are intended to be organized in a logical implementation order, but schools will consider their own needs and capacity.

Ensure students have exposure to CS each year of middle-school. It is recommended that students are enrolled in at least 1 computer science course, or a technology / engineering course that strongly incorporates computer science, per academic year.

Some typical configurations include:

- Scenario A: Year-long course, ~45 min., once per week.
- Scenario B: Semester course, ~90 min., once per week.
- Scenario C: Trimester course, ~60 min., once per week.

In an ideal situation, students will have daily exposure to CS and Engineering throughout each year of middle-school.

Ensure all HS students take at least one CS course. We recommend that all students take a ½ credit CS course in high school to fulfill their Technology requirement for graduation.This course should address each of the strands of the standards and relate CS to real-world applications.

Develop a Core CS Progression. Students with little prior exposure to computer science should take a ½ credit introductory course, as described above. Students with more experience may be able to begin HS at a higher level.

CS Progression Overview

- Introductory CS - described above.
- Intermediate CS - also addresses each of the core content areas. May include more mathematical / technical components.

Develop CS electives. Elective courses allow students to explore specific domains of computer science in greater depth. Implementation teams should consider what prerequisites may be appropriate for elective courses in CS - we don't necessarily recommend requiring the above progression before taking electives.

(See also Appendices: Examples of CS Electives for HS)

Integrate CS and develop interdisciplinary courses. We recommend that CS educators work with educators in other content areas, and professionals in other fields, to develop interdisciplinary and career connections. Integration and interdisciplinary programs of studies blur the boundaries between the disciplines.

- **Integration** refers to the inclusion of content from one content area into a course that is primarily addresses content in another area.
- **Interdisciplinary courses** combine content from one or more subject areas. Such courses should allow students to apply credit earned to either (or any) of the applicable subjects.

(See also Appendices: Interdisciplinary & Career Connections)

Leverage Career & Technical Education (CTE) programs. As previously stated, all industries are impacted by computing technology. Students will find foundational computer science knowledge and skills to be useful in a number of CTE specialty areas, including, but not limited to:

- Information and Communication Technologies
- Engineering and Manufacturing
- Health Science
- Business Management and Finance

(See also Appendices: Interdisciplinary & Career Connections.)

Appendices

Appendix A: Writers, Reviewers, & References

Standards Revision Team

The following members participated in development team meetings and/or subcommittee work.

Tammy Andrew	CS Teacher	Milford High School
Radim Bartos	CS Professor	University of New Hampshire
Heather Drolet	Tech Integrator	Christa McAuliffe School
Karen Locke	Tech Integrator, Code.org trainer	Hopkinton
Joanna Marcotte	CS Teacher	Founders Academy
Lisa Marcou	Computer Engineering Teacher	Concord Regional Technical Center
Norm Messa	CS Teacher	Seacoast School of Technology
Laura Nickerson	Director, STEM Teachers Collaborative	UNH Leitzel Center
Rajesh Prasad	CS Professor	St. Anselm College
Nancy Rose	Director of Library / Media Tech.	Merrimack School District
Mihaela Sabin	CS Professor	UNH Manchester
Zhizhang Shen	CS Professor	Plymouth State University
Alfred Thompson	CS Teacher	Bishop Guertin
Scott Valcourt	Director of Strategic Technology	University of New Hampshire
Natalya Vinogradova	Director of NH Impact Center	Plymouth State University

CS4NH Alliance advisory Committee

The following are the members of the CS4NH Alliance advisory committee.

David Benedetto	Director of STEM Education	NH Department of Education
Judy Burrows	Director of Student Aid	NH Charitable Foundation
William Church	Executive Director	White Mountain Science, Inc.
Matt Cookson	Executive Director	NH High Technology Council
Rosabel Deloge	Education Consultant	Independent
Beth Doiron	Director of DOE and College	Community College System of NH

	Access Programs	
Heather Drolet	Technology Integration Specialist	Christa McAuliffe School
Lori Langlois	Director	North Country Education Services
Laura Nickerson	Director, STEM Teachers Collaborative	University of NH, Leitzel Center
Mihaela Sabin	Computer Science Professor, Chair of Department of Engineering and Applied Sciences	University of NH, Manchester
Terry Wolf	Vice Chair, Education Committee	NH House of Representatives

Key References

NH DOE Planning Documents	• NH CS Standards Plan • NH CS State Plan
National Framework and Standards	• 2017 CSTA K-12 Standards. • K-12 CS Framework. • ISTE Standards for Students.
CS Education Data	• State-of-the-States Landscape Report on CS Education • Google-Gallup CS Polls
Computing Occupation Data	• US Bureau of Labor Statistics. STEM Occupations: Past, Present, and Future. • Change the Equation. The Hidden Half.

Glossary

- Please refer to https://k12cs.org/glossary/

Works cited

- [ACM]. Association of Computing Machinery, Computer Science Teachers Association (2010). *Running on Empty: The Failure to Teach Computer Science in the Digital Age.*
- [K12CS]. K-12 CS Coalition (2016). *K-12 Computer Science Framework.*
- [CSforALL]. US Department of Education. Office of Innovation and Improvement. Computer Science for All Proposal. Retrieved from LINK. [Link to innovation.ed.gov]

Appendix B: Useful Resources and Examples

Teaching and Learning Resources

The resources provided here are examples and are not formally endorsed by the NH Department of Education. Educators are strongly encouraged to discover and evaluate resources regularly. (See also: CS4NH Resource List.)

Primary
Resources include but are not limited to: Code.org CS Fundamentals, Project Lead the Way, Kodable, ScratchEd, Tynker, CSFirst with Google, CodeMonkey, and Khan Academy.

Secondary / Middle-Lower High School (approx 7-10 grade range)
Examples of curriculum that are appropriate for the 7-10 grade range:

- Exploring Computer Science (ECS)
- Harvey Mudd MyCS
- Code.org CS Discoveries (CSD)

Secondary / High School

- Computer Science Principles (intermediate level - can be AP or non-AP)
- AP Computer Science A (elective - algorithms & programming)

Secondary / Interdisciplinary

- Bootstrap Algebra
- Bootstrap Data Science
- Bootstrap Computational Physics

CS Electives for HS

Computing Systems / Networks & the Internet (See also: Career & Technical Education)	• Digital Electronics • Physical Computing • Cybersecurity
Algorithms & Programming (See also: Mathematics)	• Computer Programming • Data Structures • Object-Oriented Programming
Data & Analysis (See also: Mathematics)	• Data Science

Impacts of Computing (See also: Social Studies, Business, Career & Technical Education)	• Computing Career Exploration • Computer Science / Entrepreneurship • Development and Social Impacts of Information & Communication Technologies • Emerging Trends in Technology

Connections to Other Disciplines and Careers

Mathematics	
Arithmetic & Logic	• Recognizing patterns • Using number systems and representations • Arithmetic and logical operations • Developing algorithms • Developing abstractions
Algebra	• Variables, expressions, and statements • Functions
Geometry	• Using and creating computer programs to create geometric patterns and shapes
Data & Statistics	• Representing phenomena numerically and digitally • Using and creating computer simulations • Using and creating computer programs to process, analyze, and visualize data.

Sciences & Engineering	
Earth & Space Sciences	• Geographic information systems (GIS) • Agriculture and natural resource management
Physical Sciences	• Mechanics and robotics
Life Sciences	• Modeling and simulation - biological systems • Bioinformatics • Biomimicry
Engineering, Technology, & Applications of Science	• Electrical and computer engineering • Software engineering • Computational design and modeling for engineering

Visual Arts, Media Arts & Design	
Media Arts & Interactive arts (See also: Career & Technical Education)	• Audio/video production • Artbotics • Video games
Visual Arts & Design	• Computational design

Humanities & Social Sciences	
English Language Arts & World Languages	• Formal vs. natural languages • Syntax and semantics • Natural language processing • Computer translation
Social Studies	• Development and impact of information & communication technologies • Data and analytics in social sciences
Fine Arts & Performing Arts	• Computing for creative expression • Technology design / engineering for performing arts • Data and analytics for sports
Health & Wellness	• Technology use and impact on physical and mental health and wellness • Measuring and using biometrics. • Kinesiology and robotics

Career & Technical Education (CTE)	
CTE Clusters	**Examples of Computing**
STE(A)M: ● Agriculture, Food & Natural Resources ● Architecture & Construction ● Arts, A/V Technology & Communications ● Information Technology ● Health Science ● Manufacturing ● Science, Technology, Engineering & Mathematics	● Geographic Information Systems (GIS) ● Healthcare analytics ● Automated manufacturing / robotics
Business: ● Business Management & Administration ● Finance ● Hospitality & Tourism ● Marketing ● Transportation, Distribution & Logistics	● Business analytics for marketing, logistics, etc. ● Financial modeling and automation
Human Services: ● Education & Training ● Government & Public Administration ● Human Services ● Law, Public Safety, Corrections & Security	● Educational technology ● Social media ● Cybersecurity, digital forensics