Wisconsin Computer Science

Grades 9–12

Students will recognize and define computational problems using algorithms and programming AP1

A Develop algorithms. AP1.A

8 Analyze a problem and design and implement an algorithmic solution using sequence, selection, and iteration. AP1.A.8.H

9 Explain and demonstrate how modeling and simulation can be used to explore natural phenomena (e.g., flocking behaviors, queueing, life cycles). AP1.A.9.H

10 (+) Provide examples of computationally solvable problems and difficult-to-solve problems. AP1.A.10.H

11 (+) Decompose a large-scale computational problem by identifying generalizable patterns and applying them in a solution. AP1.A.11.H

12 (+) Illustrate the flow of execution of a recursive algorithm. AP1.A.12.H

13 (+) Describe how parallel processing can be used to solve large computational problems (e.g., SETI at Home, FoldIt). AP1.A.13.H

14 (+) Develop and use a series of test cases to verify that a program performs according to its design specifications. AP1.A.14.H

15 (+) Explain the value of heuristic algorithms (discovery methods) to approximate solutions for difficult-to-solve computational problems. AP1.A.15.H

Students will create computational artifacts using algorithms and programming AP2

A Develop and implement an artifact. AP2.A

10 Use user-centered research and design techniques (e.g., surveys, interviews) to create software solutions. AP2.A.10.H

11 Integrate grade-level appropriate mathematical techniques, concepts, and processes in the creation of computational artifacts. AP2.A.11.H

5 Use mathematical operations to change a value stored in a variable. AP2.A.5.I

13 (+) Decompose a computational problem by creating new data types, functions, or classes. AP2.A.13.H

14 (+) Develop programs for multiple computing platforms (e.g., computer desktop, web, mobile). AP2.A.14.H

15 (+) Implement an Artificial Intelligence (AI) algorithm to play a game against a human opponent or solve a problem. AP2.A.15.H

16 (+) Demonstrate code reuse by creating programming solutions using libraries and application program interfaces (APIs) (e.g., graphics libraries, maps, API). AP2.A.16.H

Skip to main content

Students will communicate about computing ideas AP3

A Recognize and cite sources. AP3.A

4 Compare and contrast various software licensing schemes (e.g., open source, freeware, commercial). AP3.A.4.H

B Communicate about technical and social issues. AP3.B

8 Evaluate and analyze how algorithms have impacted our society and discuss the benefits and harmful impacts of a variety of technological innovations. AP3.B.8.H

9 (+) Compare a variety of programming languages and identify features that make them useful for solving different types of problems and developing different kinds of systems (e.g., declarative, logic, parallel, functional, compiled, interpreted, real-time). AP3.B.9.H

10 (+) Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality). AP3.B.10.H

C Document code. AP3.C

3 (+) Describe how Artificial Intelligence (AI) drives many software and physical systems (e.g., autonomous robots, computer vision, pattern recognition, text analysis). AP3.C.3.H

4 Write appropriate documentation for programs. AP3.C.4.H

5 (+) Use application programming interface (APIs) documentation resources. AP3.C.5.H

6 Use online resources to answer technical questions. AP3.C.6.H

Students will develop and use abstractions AP4

A Create and use abstractions (representations) to solve complex computational problems. AP4.A

4 Demonstrate the value of abstraction for managing problem complexity (e.g., using a list instead of discrete variables). AP4.A.4.H

5 Understand the notion of hierarchy and abstraction in high-level languages, translation, instruction sets, and logic circuits. AP4.A.5.H

6 Deconstruct a complex problem into simpler parts using predefined constructs (e.g., functions and parameters and/or classes). AP4.A.6.H

7 (+) Compare and contrast fundamental data structures and their uses (e.g., lists, maps, arrays, stacks, queues, trees, graphs). AP4.A.7.H

8 (+) Critically analyze and evaluate classic algorithms (e.g., sorting, searching) and use in different contexts, adapting as appropriate. AP4.A.8.H

9 (+) Discuss issues that arise when breaking large-scale problems down into parts that must be processed simultaneously on separate systems (e.g., cloud computing, parallelization, concurrency). AP4.A.9.H

10 (+) Define the functionality of an abstraction without implementing the abstraction. AP4.A.10.H

Skip to main content

11 (+) Evaluate algorithms (e.g., sorting, searching) in terms of their efficiency, correctness, and clarity. AP4.A.11.H

12 (+) Identify programming language features that can be used to define or specify an abstraction. AP4.A.12.H

13 (+) Identify abstractions used in a solution (program or software artifact) and reuse those abstractions to solve a different problem. AP4.A.13.H

Students will collaborate with diverse teams AP5

A Work together to solve computational problems using a variety of resources. AP5.A

6 Design and develop a software artifact working in a team. AP5.A.6.H

7 Demonstrate how diverse collaborating impacts the design and development of software products (e.g., discussing real-world examples of products that have been improved through having a diverse design team or reflecting on their own team's development experience). AP5.A.7.H

8 (+) Demonstrate software life cycle processes (e.g., spiral, waterfall) by participating on software project teams (e.g., community service project with real-world clients). AP5.A.8.H

9 (+) Use version control systems, integrated development environments (IDEs), and collaboration tools and practices (code documentation) in a group software project. AP5.A.9.H

B Foster an inclusive computing culture. AP5.B

3 Create design teams taking into account the strengths and perspectives of potential team members. AP5.B.3.H

Students will test and refine computational solutions AP6

A Test and debug computational solutions. AP6.A

4 Use a systematic approach and debugging tools to independently debug a program (e.g., setting breakpoints, inspecting variables with a debugger). AP6.A.4.H

B Develop and apply success criteria. AP6.B

3 (+) Evaluate key qualities of a program (e.g., correctness, usability, readability, efficiency, portability, scalability) through a process such as a code review. AP6.B.3.H

Students will communicate about computing systems CS1

A Identify hardware and software components. CS1.A

6 Develop and apply criteria (e.g., power consumption, processing speed, storage space, battery life, cost, operating system) for evaluating a computer system for a given purpose (e.g., system specification needed to run a game, web browsing, graphic design, or video editing). CS1.A.6.H

7 (+) Identify the functionality of various categories of hardware components and communication between them (e.g., physical layers, logic gates, chips, input and output devices). CS1.A.7.H

Understand how the components of a computer system work together. CS1.B

Skip to main content

3 (+) Explain the role of operating systems (e.g., how programs are stored in memory, how data is organized and retrieved, how processes are managed and multi-tasked). CS1.B.3.H

Students will test and refine computing systems CS2

A Problem solve and debug. CS2.A

4 Devise a systematic process to identify the source of a problem within individual and connected devices (e.g., research, investigate, problem solve). CS2.A.4.H

Students will develop and use abstractions in computing systems CS3

A Generalize in computer systems. CS3.A

2 Demonstrate the role and interaction of a computer embedded within a physical system, such as a consumer electronic, biological system, or vehicle, by creating a diagram, model, simulation, or prototype. CS3.A.2.H

3 (+) Describe the steps necessary for a computer to execute high-level source code (e.g., compilation to machine language, interpretation, fetch-decode-execute cycle). CS3.A.3.H

Students will create and modify computing systems CS4

A Modify and create computational artifacts. CS4.A

2 Create, extend, or modify existing programs to add new features and behaviors using different forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds). CS4.A.2.H

3 (+) Create a new artifact that uses a variety of forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds). CS4.A.3.H

Students will create computational artifacts using data and analysis DA1

A Represent and manipulate data. DA1.A

4 Convert between binary, decimal, and hexadecimal representations of data (e.g., convert hexadecimal color codes to decimal percentages, ASCII/Unicode representation). DA1.A.4.H

5 Analyze the representation tradeoffs among various forms of digital information (e.g., lossy vs. lossless compression, encrypted vs. unencrypted, various image representations). DA1.A.5.H

6 (+) Discuss how data sequences (e.g., binary, hexadecimal, octal) can be interpreted in a variety of forms (e.g., instructions, numbers, text, sound, image). DA1.A.6.H

Students will recognize and define data in computational problems DA2

A Gather data to support computational problem solving. DA2.A

4 Discuss techniques used to store, process, and retrieve different amounts of information (e.g., files, databases, data warehouses). DA2.A.4.H

5 (+) Use various data collection techniques for different types of computational problems (e.g., mobile device Global Positioning System (GPS), user surveys, embedded system sensors, open data sets, social media data sets). DA2.A.5.H

Skip to main content

:ategorize and analyze data. DA2.B

4 Apply basic techniques for locating and collecting small- and large-scale data sets (e.g., creating and distributing user surveys, accessing real-world data sets). DA2.B.4.H

Students will communicate about data in computing DA3

A Communicate about data. DA3.A

6 Use computational tools to collect, transform, and organize data about a problem to explain to others. DA3.A.6.H

Students will develop and use data abstractions DA4

A Model with data. DA4.A

6 Create computational models that simulate real-world systems (e.g., ecosystems, epidemics, spread of ideas). DA4.A.6.H

7 (+) Evaluate the ability of models and simulations to formulate, refine, and test hypotheses. DA4.A.7.H

B Identify patterns. DA4.B

1 (+) Use data analysis to identify significant patterns in complex systems (e.g., take existing data sets and make sense of them). DA4.B.1.H

2 (+) Identify mathematical and computational patterns through modeling and simulation (e.g., regression, queueing theory, discrete event simulation). DA4.B.2.H

2 (+) Identify mathematical and computational patterns through modeling and simulation (e.g., regression, queueing theory, discrete event simulation). DA4.B.2.H

Students will understand the impact and effect computing technology has on our everyday lives IC1

A Understand the impact technology has on our everyday lives and the effects of computing on the economy and culture. IC1.A

6 Debate the social and economic implications associated with ethical and unethical computing practices (e.g., intellectual property rights, hacktivism, software piracy, new computers shipped with malware). IC1.A.6.H

7 Discuss implications of the collection and large-scale analysis of information about individuals (e.g., how businesses, social media, and government collect and use personal data). IC1.A.7.H

8 Compare and debate the positive and negative impacts of computing on behavior and culture (e.g., evolution from hitchhiking to ride-sharing apps, online accommodation rental services). IC1.A.8.H

9 Describe how computation shares features with art and music by translating human intention into an artifact. IC1.A.9.H

10 (+) Develop criteria to evaluate the beneficial and harmful effects of computing innovations on people and society. IC1.A.10.H

B Understand the effects of computing on communication and relationships. IC1.B

5 Evaluate the negative impacts of electronic communication on personal relationships and evaluate differences between face-to-face and electronic communication. IC1.B.5.H

Skip to main content

6 (+) Create a list of practices that individuals and organizations can use to encourage proper use of both electronic and face-to-face communication. IC1.B.6.H

7 (+) Evaluate the negative impacts on societal discourse caused by social media and electronic communities. IC1.B.7.H

Students will experience learning within a collaborative, inclusive computing culture and explain the steps needed to ensure that all people have access to computing IC2

A Understand the effects of the digital divide. IC2.A

3 (+) Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society. IC2.A.3.H

B Test and refine digital artifacts for accessibility. IC2.B

3 Design a user interface (e.g., web pages, mobile applications, animations) to be more inclusive and accessible, minimizing the impact of the designer's inherent bias. IC2.B.3.H

C Collaborate ethically in the creation of digital artifacts. IC2.C

5 Ethically and safely select, observe, and contribute to global collaboration in the development of a computational artifact (e.g., contribute the resolution of a bug in an open-source project platform, or contribute an online article). IC2.C.5.H

6 Demonstrate how computing enables new forms of experience, expression, communication, and collaboration. IC2.C.6.H

Students will understand the importance of proper use of data and information in a computing society IC3

A Understand intellectual property and fair use. IC3.A

4 Compare and contrast information access and distribution rights. IC3.A.4.H

B Assess the practice of digital privacy. IC3.B

5 Research and understand misuses of private digital information in our society. IC3.B.5.H

6 Debate laws regarding an individual's digital privacy and be able to explain the main arguments from multiple perspectives. IC3.B.6.H

C Assess interrelationship between computing and society. IC3.C

1 (+) Design and implement a study that evaluates how computation has revolutionized an aspect of our culture or predicts how an aspect might evolve (e.g., education, healthcare, art/entertainment, energy). IC3.C.1.H

2 (+) Debate laws and regulations that impact the development and use of software and be able to explain the main arguments from multiple perspectives. IC3.C.2.H

Students will understand the importance of security when using tec

Skip to main content

A Use secure practices for personal computing. NI1.A

6 Provide examples of personal data that should be kept secure and the methods by which individuals keep their private data secure. NI1.A.6.H

7 (+) Explain security issues that might lead to compromised computer programs (e.g., circular references, ambiguous program calls, lack of error checking, and field size checking). NI1.A.7.H

B Understand the importance of institutional security. NI1.B

3 Compare and contrast multiple viewpoints on cybersecurity (e.g., from the perspective of security experts, privacy advocates, national security). NI1.B.3.H

4 Identify digital and physical strategies to secure networks and discuss the tradeoffs between ease of access and need for security. NI1.B.4.H

Students will understand how information is sent by the internet NI2

A Demonstrate how the internet works at the physical layer. NI2.A

8 Illustrate the basic components of computer networks (e.g., draw logical and topological diagrams of networks including routers, switches, servers, and end user devices; create model with string and paper). NI2.A.8.H

9 (+) Explain ways in which the internet is decentralized and fault-tolerant. NI2.A.9.H

10 (+) Simulate and discuss the issues (e.g., bandwidth, load, delay, topology) that impact network functionality (e.g., use free network simulators). NI2.A.10.H

B Demonstrate how the internet works at the protocol layer. NI2.B

3 Describe key protocols and underlying processes of internet-based services (e.g., http/https and Simple Mail Transfer Protocol (SMTP) or Internet Message Access Protocol (IMAP), routing protocols). NI2.B.3.H

C Demonstrate how the internet works at the addressing layer. NI2.C

4 (+) Evaluate how the hierarchical nature of the Domain Name System helps the internet work efficiently. NI2.C.4.H

D Demonstrate and explain encryption methods. NI2.D

3 Write a program that performs basic encryption (e.g., shift cipher, substitution cipher). NI2.D.3.H

4 (+) Explain the features of public key cryptography. NI2.D.4.H

5 (+) Explore security policies by implementing and comparing encryption and authentication strategies (e.g., secure coding, safeguarding keys). NI2.D.5.H