Indiana K-12 Computer Science Standards

What is Computer Science?

Computer science is “the study of computers and algorithmic processes, including their principles, their hardware and software designs, their [implementation], and their impact on society” (Tucker et. al, 2003, p. 6).

Computer science has also been defined as “the study of computers and ALL the phenomena that arise around them” (Herbert Simon).

Indiana’s Standards

Indiana’s Academic Standards for Computer Science allow for students to be prepared in the everchanging computer science areas providing inquiry-based, hands-on experiences based on two components: Concepts and Practices. K-8 standards have been in place and implemented since 2016 and high school course standards have existed for some time. Particularly in K-8, computer science standards can be integrated into various other subject areas. A variety of elective courses are available for high schools. As students move through grade levels, they will work with and experience the standards at those grade bands (K-2, 3-5, 6-8, and 9-12). The standards are based on the following core concepts and core practices:

Indiana’s Five Core Computer Science Concepts:

1. Data and Information
2. Computing Devices and Systems
3. Programs and Algorithms
4. Networking and Communication
5. Impact and Culture

Nationally Recognized Core Computer Science Practices from the K-12 CS Framework:

1. Fostering an Inclusive Computing Culture
2. Collaborating Around Computing
3. Recognizing and Defining Computational Problems
4. Developing and Using Abstractions
5. Creating Computational Artifacts
6. Testing and Refining Computational Artifacts
7. Communicating About Computing

Color key:

Indicates K-8 Standards

Indicates 9-12 Standards

Standards prefix key:

K-2: Grades K-2 grade band standard

3-5: Grades 3-5 grade band standard

6-8: Grades 6-8 grade band standard

ICS: Introduction to Computer Science

CSI: Computer Science I

CSII: Computer Science II

K-12 CS Standards - Data and Information	
K-2.DI.1	Use technology resources to solve age-appropriate problems and communicate thoughts, ideas, or stories in a step-by-step manner.
K-2.DI.2	Understand how to arrange (sort) information into useful order, such as sorting students by birth date, without using a computer.
K-2.DI.3	Recognize that software is created to control computer operations.
3-5.DI.1	Understand and use the basic steps in algorithmic problem solving (e.g., problem statement and exploration, examination of sample instances, design, implementation, and testing).
3-5.DI.2	Develop a simple understanding of an algorithm (e.g., search, sequence of events, or sorting) using computer-free exercises.
3-5.DI.3	Demonstrate how a string of bits can be used to represent alphanumeric information and how 1's and 0's represent information.
3-5.DI.4	Describe how a simulation can be used to solve a problem.
3-5.DI.5	Understand the connections between computer science and other fields.
6-8.DI.1	Use the basic steps in algorithmic problem-solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing, and evaluation).
6-8.DI.2	Describe the process of parallelization as it relates to problem solving.
6-8.DI.3	Represent data in a variety of ways (e.g., text, sounds, pictures, and numbers), and use different visual representations of problems, structures, and data (e.g., graphs, charts, network diagrams, flowcharts).

6-8.DI.4	Understand the notion of hierarchy and abstraction in computing including high-level languages, translation, instruction set, and logic circuits.
6-8.DI.5	Demonstrate interdisciplinary applications of computational thinking and interact with content-specific models and simulations to support learning and research.
ICS-2.1	Use the design process to iteratively develop a computing artifact.
ICS-2.3	Understand how abstractions hide implementation details when used in everyday objects.
ICS-2.4	Use abstraction to manage program complexity (such as a function to create recallable code).
ICS-2.5	Formulate algorithms using programming structures to decompose a complex problem.
ICS-3.1	Understand how computers represent data, including text, sound, images, and numbers.
ICS-3.2	Create data visualizations, models, and simulations.
ICS-3.3	Evaluate data to better understand the world.
ICS-3.4	Explore the relationship between information and data.
ICS-5.3	Utilize a problem solving approach to develop a solution using technology.
CSI-3.7	Describe the different methods for encoding data such as binary, decimal, hexadecimal, ASCII, and Unicode.
CSI-6.1	Describe the function of a computing artifact (for example, code or design).
CSI-6.2	Identify the purposes of a computing artifact.
CSI-6.3	Explain concepts related to a computing artifact.
CSI-6.4	Describe how to use a computing artifact.
CSI-6.5	Explain cause/effect by interpreting input and output.
CSI-6.6	Create documentation for computing artifact such as comments or user manual/readme.
CSII-3.11	Demonstrate the different methods for encoding data such as binary, decimal, hexadecimal, ASCII, and Unicode.
CSII-6.1	Describe the function of a computing artifact (for example, code or design).
CSII-6.2	Identify the purposes of a computing artifact.

CSII-6.3	Explain concepts related to a computing artifact.
CSII-6.4	Describe how to use a computing artifact.
CSII-6.5	Explain cause/effect by interpreting input and output.
CSII-6.6	Create documentation for computing artifact such as comments or user manual/readme.
Additional standards possible dependent upon selected capstone course.	

K-12 Computer Science - Computing Devices and Systems	
K-2.CD.1	Use standard input and output devices to operate computers and other technologies.
3-5.CD.1	Demonstrate proficiency with keyboards and other input and output devices.
3-5.CD.2	Understand the pervasiveness of computers and computing in daily life (e.g., voicemail, downloading videos and audio files, microwave ovens, thermostats, wireless Internet, mobile computing devices, GPS systems).
3-5.CD.3	Apply troubleshooting strategies for identifying simple hardware and software problems that may occur during use.
3-5.CD.4	Recognize that computers model intelligent behavior (as found in robotics, speech and language recognition, and computer animation).
6-8.CD.1	Demonstrate an understanding of the relationship between hardware and software.
6-8.CD.2	Apply troubleshooting strategies to identify and solve routine hardware and software problems that occur during everyday computer use.
6-8.CD.3	Describe the major components and functions of computer systems and network.
6-8.CD.4	Describe what distinguishes humans from machines focusing on human intelligence versus machine intelligence and ways we can communicate, as well as ways in which computers use models of intelligent behavior (e.g., robot motion, speech and language understanding, and computer vision).
ICS-2.6	Assess a program by testing to verify correct behavior.
ICS-4.1	Demonstrate understanding of the hardware and operating systems of computers.
ICS-4.3	Explore the fundamental principles and components of computer networking.

ICS-4.5	Investigate the use of artificial intelligence by individuals and society.
ICS-4.6	Investigate innovations in computing, including robotics.
Additional standards possible dependent upon selected capstone course.	

K-12 Computer Science - Programs and Algorithms	
K-2.PA.1	Use technology and developmentally appropriate multimedia resources to conduct age-appropriate research and support learning across the curriculum.
K-2.PA.2	Create developmentally appropriate multimedia products with support from teachers, family members, or student partners.
K-2.PA.3	Arrange information using concept mapping tools and a set of statements that accomplish a simple task.
3-5.PA.1	Use technology resources (e.g., calculators, data collection probes, mobile devices, videos, educational software, and web tools) for problem-solving and self-directed learning, and general-purpose productivity tools and peripherals to support personal productivity, remediate skill deficits, facilitate learning, and individual/collaborative writing, communication, and publishing activities.
3-5.PA.2	Use digital tools to gather, manipulate, and modify data for use by a program.
3-5.PA.3	Implement problem solutions using a block-based visual programming language.
6-8.PA.1	Select appropriate tools and technology resources to support learning and personal productivity, publish individual products, and design, develop, and publish data, accomplish a variety of tasks, and solve problems.
6-8.PA.2	Implement problem solutions using a programming language that includes looping behavior, conditional statements, logic, expressions, variables, and functions.
6-8.PA.3	Demonstrate dispositions amenable to open-ended problem solving and programming (e.g., comfort with complexity, persistence, brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge).
ICS-2.2	Demonstrate competencies of programming constructs, including: use of data types and variables, control structures (sequencing, looping, branching), and modularity (such as a function).
ICS-2.7	Construct a computing artifact that has a user interface.
ICS-2.8	Produce an artifact that includes rich media.

ICS-2.9	Illustrate knowledge of good programming practice including the use of conventional standards and comment.
ICS-5.5	Program a solution to a problem using pair programming or other methods.
CSI-1.1	Document problem analysis through industry standards such as: flowcharts, functional specifications, user stories, etc.
CSI-1.2	Outline the problem assigned and describe the solution.
CSI-1.3	Use puzzles and games to enhance problem solving skills.
CSI-1.4	Recognize language appropriate planning and designing tools (for example: flowcharts, UML diagrams, pseudocode, use cases).
CSI-3.1	Develop algorithms to determine a solution.
CSI-3.2	Assess the use of algorithms to provide a solution.
CSI-3.3	Use pseudocode to describe a solution.
CSI-3.4	Create a program flowchart using ANSI standard flowcharting symbols to define a solution.
CSI-3.5	Explain how the algorithm can be used to solve a problem.
CSI-3.6	Apply Boolean logic and relational operations.
CSI-4.1	Define the process of programming.
CSI-4.2	Create a computer program that corresponds to an algorithm or proposed solution.
CSI-4.3	Define fundamental control structures (sequencing, selection, and repetition).
CSI-4.4	Implement data variables and constants.
CSI-4.5	Compare local scope and global scope.
CSI-4.6	Construct a fundamental control structure.
CSI-4.7	Implement arrays.
CSI-4.8	Create programmer defined functions and methods to break down program logic and support reuse.
CSI-4.9	Compare the graphical user interface and the command line interface.
CSI-4.10	Recognize the order of operations used by a computer when performing calculations.

CSI-4.11	Define simple and compound conditionals (Boolean - not, and, or).
CSI-4.12	Implement simple and compound conditionals.
CSI-4.13	Adhere to industry standard programming conventions for accuracy and readability.
CSI-5.1	Predict and explain programming outcomes.
CSI-5.2	Identify cause/effect for input/output.
CSI-5.3	Understand input validation.
CSI-5.5	Conduct testing to identify performance errors.
CSI-5.6	Differentiate between syntax and logic errors.
CSI-5.7	Debug code using techniques such as: code tracing, print statements, boundary testing, breakpoints, etc.
CSII-1.1	Document problem analysis through industry standards such as: flowcharts, functional specifications, user stories, etc.
CSII-1.2	Outline the problem assigned and describe the solution.
CSII-1.3	Use puzzles and games to enhance problem solving skills.
CSII-1.4	Recognize language appropriate planning and designing tools (for example: flowcharts, UML diagrams, pseudocode, use cases).
CSII-1.5	Modify an existing program, such as a template, to add additional functionality and discuss intended and unintended implications.
CSII-1.6	Justify what programming methodology to use (object oriented or procedural).
CSII-3.1	Develop algorithms to determine a solution.
CSII-3.2	Assess the use of algorithms to provide a solution.
CSII-3.3	Create pseudocode to describe a solution.
CSII-3.4	Create a program flowchart using ANSI standard flowcharting symbols to define a solution.
CSII-3.5	Explain how the algorithm can be used to solve a problem.
CSII-3.6	Apply Boolean logic and relational operations.
CSII-3.7	Demonstrate iteration in an algorithm.

CSII-3.8	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
CSII-3.9	Illustrate the flow of execution of a recursive algorithm.
CSII-3.10	Integrate classic algorithms (sorting and searching) to solve computational problems.
CSII-4.1	Define the process of programming.
CSII-4.2	Plan and develop programs for broad audiences using a software development lifecycle process (such as: Agile, waterfall, and spiral).
CSII-4.3	Adhere to industry standard programming conventions for accuracy and readability.
CSII-4.4	Demonstrate code reuse by creating programming solutions using libraries and APIs.
CSII-4.5	Create an advanced computer program that corresponds to an algorithm or proposed solution.
CSII-4.6	Implement fundamental programming constructs, including data types, control structures, methods, and arrays.
CSII-4.7	Compare local scope and global scope.
CSII-4.8	Create programmer defined functions and methods to break down program logic and support reuse.
CSII-4.9	Recognize the order of operations used by a computer when performing calculations.
CSII-4.10	Implement simple and compound conditionals (Boolean - not, and, or).
CSII-4.11	Demonstrate the relationship between classes and objects (instances).
CSII-4.12	Demonstrate the differences between instance variables and class variables.
CSII-4.13	Demonstrate the differences between instance methods and class methods.
CSII-4.14	Apply inheritance, polymorphism, encapsulation, and abstraction in the context of Object Oriented Programming.
CSII-4.15	Compare the graphical user interface and the command line interface.
CSII-5.1	Predict and explain programming outcomes.
CSII-5.2	Identify cause/effect for input/output.
CSII-5.3	Understand input validation.

CSII-5.4	Develop and use a series of test cases to verify that a program performs according to its design specifications.
CSII-5.5	Peer review documentation and code.
CSII-5.6	Differentiate between syntax, logic, and semantic errors.
CSII-5.7	Debug code using techniques such as: code tracing, print statements, boundary testing, breakpoints, etc.
CSII-5.8	Review computing artifacts to reduce bias, increase equity, and support accessibility.
Additional standards possible dependent upon selected capstone course.	

K-12 Computer Science - Networking and Communication	
K-2.NC.1	Use technology to work cooperatively and collaboratively with peers, teachers, and others.
K-2.NC.2	Gather information and communicate electronically with others with support from teachers, family members, or student partners.
3-5.NC.1	Use online resources (e.g., email, online discussions, collaborative web environments) to participate in collaborative problem-solving activities for the purpose of developing solutions or products.
3-5.NC.2	Use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual and collaborative writing, communication, and publishing activities.
6-8.NC.1	Collaboratively design, develop, publish, and present products (e.g., videos, podcasts, websites) using technology resources that demonstrate and communicate curriculum concepts.
6-8.NC.2	Exhibit dispositions necessary for collaboration: providing useful feedback, integrating feedback, understanding and accepting multiple perspectives, socialization.
ICS-5.1	Design a solution to a problem by working in a team.
ICS-5.2	Explore technologies that can be used to collaborate with others of various cultures and career fields.
ICS-5.4	Analyze the work of peers and provide feedback.
CSI-2.1	Design a solution to a problem by working in a team.

CSI-2.2	Explore technologies that can be used to collaborate with others of various cultures and career fields.
CSI-2.3	Utilize a problem solving approach to develop a solution using technology.
CSI-2.4	Analyze the work of peers and provide feedback.
CSI-2.5	Program a solution to a problem using pair programming or other methods.
CSI-5.4	Peer review documentation and code.
CSII-2.1	Design a solution to a problem by working in a team.
CSII-2.2	Explore technologies that can be used to collaborate with others of various cultures and career fields.
CSII-2.3	Utilize a problem solving approach to develop a solution using technology.
CSII-2.4	Analyze the work of peers and provide feedback.
CSII-2.5	Use version control systems, integrated development environments (IDEs), and collaborative tools and practices in a group software project.
Additional standards possible dependent upon selected capstone course.	

K-12 Computer Science - Impact and Culture	
K-2.IC.1	Practice responsible digital citizenship (legal and ethical behaviors) in the use of technology.
K-2.IC.2	Identify positive and negative social and ethical behaviors for using technology.
3-5.IC.1	Discuss basic issues related to responsible use of technology and information, and the consequences of inappropriate use.
3-5.IC.2	Identify the impact of technology (e.g., social networking, cyber bullying, mobile computing and communication, web technologies, cyber security, and virtualization) on personal life and society.
3-5.IC.3	Evaluate the accuracy, relevance, appropriateness, comprehensiveness, and biases that occur in electronic information sources.
3-5.IC.4	Understand ethical issues that relate to computers and networks (e.g., equity of access, security, privacy, copyright, and intellectual property).
6-8.IC.1	Exhibit legal and ethical behaviors when using technology and information and discuss the consequences of misuse.

6-8.IC.2	Analyze the positive and negative impacts of technology on one's personal life, society, and our culture.
6-8.IC.3	Evaluate the accuracy, relevance, appropriateness, comprehensiveness, and biases that occur in electronic information sources.
6-8.IC.4	Describe ethical issues that relate to computers and networks (e.g., security, privacy, ownership, and information sharing), and discuss how unequal distribution of technological resources in a global economy raises issues of equity, access, and power.
ICS-1.1	Create a definition of computer science and computational thinking.
ICS-1.2	Summarize ethical issues within computer science.
ICS-1.3	Investigate trends in computer science and their impact on society.
ICS-1.4	Summarize ethical issues within computer science.
ICS-4.2	Discuss the ethical and appropriate use of computer devices.
ICS-4.4	Examine the impact of the Internet on society.
ICS-6.1	Examine the dynamic between privacy and security.
ICS-6.2	Explain the privacy concerns related to the collection and generation of data through implicit and explicit processes.
ICS-6.3	Evaluate the social and emotional implications of privacy in the context of safety, law, and ethics.
ICS-6.4	Give examples to illustrate how sensitive data can be affected by malware and other attacks.
ICS-6.5	Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical implications.
ICS-6.6	Discuss the laws surrounding intellectual property.
ICS-7.1	Identify computer science occupations and the roles and responsibilities of each.
ICS-7.2	Report job outlook, demand, and projected wages for computer science careers.
ICS-7.3	Explore the job opportunities that are available in computer science.
ICS-7.4	Investigate post-secondary training opportunities and industry certifications that are available.

CSI-5.8	Review computing artifacts to reduce bias, increase equity, and support accessibility.
CSI-7.1	Examine the dynamic between privacy and security.
CSI-7.2	Explain the privacy concerns related to the collection and generation of data through implicit and explicit processes.
CSI-7.3	Evaluate the social and emotional implications of privacy in the context of safety, law, and ethics.
CSI-7.4	Give examples to illustrate how sensitive data can be affected by malware and other attacks.
CSI-7.5	Discuss the concepts and justifications for using secure design techniques.
CSI-7.6	Discuss the laws surrounding intellectual property.
CSI-8.1	Identify computer science occupations and the roles and responsibilities of each.
CSI-8.2	Report job outlook, demand, and projected wages for computer science careers.
CSI-8.3	Explore the job opportunities that are available in computer science.
CSI-8.4	Investigate post-secondary training opportunities and industry certifications that are available.
CSII-7.1	Examine the dynamic between privacy and security.
CSII-7.2	Explain the privacy concerns related to the collection and generation of data through implicit and explicit processes.
CSII-7.3	Evaluate the social and emotional implications of privacy in the context of safety, law, and ethics.
CSII-7.4	Give examples to illustrate how sensitive data can be affected by malware and other attacks.
CSII-7.5	Discuss the concepts and justifications for using secure design techniques.
CSII-7.6	Discuss the laws surrounding intellectual property.
CSII-8.1	Identify computer science occupations and the roles and responsibilities of each.
CSII-8.2	Report job outlook, demand, and projected wages for computer science careers.

CSII-8.3	Explore the job opportunities that are available in computer science.
CSII-8.4	Investigate post-secondary training opportunities and industry certifications that are available.
Additional standards possible dependent upon selected capstone course.	