2023 Computer Science (Emended Feb. 2025)

Wyoming Content & Performance Standards (WYCPS)

Effective - July 17, 2024

To be Fully Implemented in Districts by the Beginning of School Year 2026-27

Rationale:

Computing is fundamental to understanding and participating in an increasingly technological society, and is essential for every Wyoming student to learn as part of a modern education. Computer science is a subject that provides students with a critical lens for interpreting the world around them and challenges them to explore how computing and technology can expand Wyoming's impact on the world.

The standards present here provide the necessary foundation for local school district decisions about curriculum, instruction, and assessment. Implementation of these standards will better prepare Wyoming high school graduates for the rigors of college and/or career. In turn, Wyoming employers will be able to hire workers with a strong foundation in Computer Science—both in specific content areas and in critical thinking and inquiry-based problem solving.

The Computer Science Standards are written in grade bands (K-2, 3-5, 6-8, and 9-12).

Organization of the Standards:

Standard Code: Grade.Domain.Subconcept.Standard#

Key: 2.CS.HS.01 = 2nd Grade.Computing Systems.Hardware and Software.Standard #1

Domain:

The core concepts to be studied in computer science are as follows: 1) Computing Systems; 2) Networks and the Internet; 3) Data and Analysis; 4) Algorithms and Programming; and 5) Impacts of Computing.

Plugged in:

This symbol designates when a standard may require hardware, software, or both in order to fully address the intent of the standard.

Computer Science:

Computer Science is the study of computing principles, design, and applications (hardware & software); the creation, access, and use of information through algorithms and problem solving, and the impact of computing on society.

Computational Thinking:

Computational thinking is a necessary and meaningful 21st century skill. Computational thinking is defined as the thought processes involved in formulating a problem and expressing its solutions in such a way that a computer (human or machine) can effectively carry them out. Computational thinking develops into competencies in problem solving, critical thinking, productivity, and creativity. Over time, engaging in computational thought builds a student's capacity to persevere, work efficiently, gain confidence, recognize and resolve ambiguity, generalize concepts, and communicate effectively. In order to adapt to global advancements in technology, students will need to use their computational thinking skills to formulate, articulate, and discuss solutions in a meaningful manner.

Computer Science (CS) Practices:
There are seven (7) CS Practices that can be used as the standards are taught and measured. The seven (7) CS Practices are listed below:

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

Grade K-2 Computer Science Content Standards

Introduction:
K-2 Students may be most familiar with touch devices. These students may not yet understand the use of computing devices beyond playing games. They may have emerging problem-solving skills and introductory level sequencing abilities, but their understanding of programming concepts may be limited.

Computing Systems

Devices (D), Hardware & Software (HS), and Troubleshooting (T)

2.CS.HS.01 Demonstrate and describe the function of common components of computing systems (hardware and software) (e.g., use a browser, search engine). [Practice 7.2 Communicating About Computing]

Network & The Internet

Network, Communication, & Organization (NCO) and Cybersecurity (C)

2.NI.C.01 Explain what authentication factors (e.g., login) are, why we use them, and apply authentication to protect devices and information (personal and private) from unauthorized access. [Practice 7.3 Communication About Computing]

Algorithms & Programming

Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

2.AP.A.01 With guidance, identify and model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks (e.g., verbally, kinesthetically, with robot devices, or a programming language). [Practice 4.4 Developing and Using Abstractions]

2.AP.C.01 With guidance, independently and collaboratively create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing, conditionals, and repetition. [Practice 5.2 Creating Computational Artifacts]

2.AP.PD.03 Independently and collaboratively debug (identify and fix errors) programs using a programming language. [Practice 6.2 Testing and Refining Computational Artifacts]

Impacts of Computing

Culture (C) and Social Interactions (SI)

2.IC.SI.01 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [Practice 2.1 Collaborating Around Computing]

3-5 Computer Science Content Standards

Introduction:
Throughout grades 3-5, students engage in creative applications of Computer Science concepts and practices introduced in K-2. By the end of fifth grade, students will build upon their previous understanding of algorithms, programming (coding), networks, and the Internet. In addition, students will create, modify, and troubleshoot increasingly complex programs for a variety of purposes. Students will be able to explain cultural, social, and ethical impacts of computing.

Computing Systems

Devices (D), Hardware & Software (HS), and Troubleshooting (T)

5.CS.HS.01 Model how information is translated, transmitted, and processed in order to follow through hardware and software to accomplish tasks. [Practice 4.4 Developing and Using Abstractions]

Network & The Internet

Network, Communication, & Organization (NCO) and Cybersecurity (C)

5.NI.C.01 Discuss real-world cybersecurity problems and identify and implement appropriate strategies for how personal information can be protected. [Practice 3.1 Recognizing and Defining Computational Problems]

Data Analysis

Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

5.DA.S.01 Justify the format and location for storing data based on sharing requirements and the type of information (e.g., images, videos, text). [Practice 4.2 Developing and Using Abstractions]

Algorithms & Programming

Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

5.AP.A.01 Using grade appropriate content and complexity, compare and refine multiple algorithms for the same task and determine which is the most appropriate. [Practice 3.3 Recognizing and Defining Computational Problems] [Practice 6.3 Testing and Refining Computational Artifacts]

5.AP.C.01 Using grade appropriate content and complexity, create programs that include sequences, events, loops, and conditionals, both individually and collaboratively. [Practice 5.2 Creating Computational Artifacts]

Impacts of Computing

Culture (C), Social Interactions (SI), and Safety, Law, and Ethics (SLE)

5.IC.SI.02 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [Practice 2.1 Collaborating Around Computing]

6-8 Computer Science Content & Performance Standards

Introduction:
Throughout grades 6-8, students continue to develop their understanding of algorithms and programming (coding). Students work collaboratively and independently to create and modify increasingly complex programs for a variety of purposes introduced in grades 3-5.

Computing Systems

Devices (D), Hardware & Software (HS), and Troubleshooting (T)

8.CS.HS.01 Design and refine a project that combines hardware and software components to collect and exchange data. [Practice 5.1 Creating Computational Artifacts]

- The Proficient student:
  - Designs a project that combines hardware and software components to collect and exchange data.
  - Refines a project that combines hardware and software components to collect and exchange data.

Network & The Internet

Network, Communication, & Organization (NCO) and Cybersecurity (C)

8.NI.C.01 Critique physical and digital procedures that could be implemented to protect electronic data/information. [Practice 7.3 Communicating About Computing]

- The Proficient student critiques physical and digital procedures that could be implemented to protect electronic data/information.

Data Analysis

Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

8.DA.CVT.01 Using computational tools, transform collected data to make it more useful and reliable. [Practice 6.3 Testing and Refining Computational Artifacts]

- The Proficient student uses computational tools to:
  - Transform data to improve reliability by removing errors.
  - Highlight or expose relationships in the data.

Algorithms & Programming

Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

8.AP.V.01 Using grade appropriate content and complexity, create clearly named variables that represent different data types and perform operations on their values. [Practice 5.1 & 5.2 Creating Computational Artifacts]

- The Proficient student:
  - Creates clearly named variables.
  - Creates variables that represent different data types.
  - Performs operations on the values of variables.

8.AP.C.01 Using grade appropriate content and complexity, design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. [Practice 5.1 & 5.2 Creating Computational Artifacts]

- The Proficient student designs and iteratively develops programs that include:
  - Nested loops.
  - Compound conditionals.

8.AP.M.01 Using grade appropriate content and complexity, decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. [Practice 3.2 Recognizing and Defining Computational Problems]

- The Proficient student decomposes problems and subproblems into parts for the design, implantation, and review of programs.

8.AP.PD.04 Using grade appropriate content and complexity, document programs in order to make them easier to follow, test, and debug. [Practice 7.2 Communicating About Computing]

- The Proficient student documents programs in order to make them easier to follow, test, and debug.

Impacts of Computing

Culture (C), Social Interactions (SI), and Safety, Law, and Ethics (SLE)

8.IC.C.01 Describe impacts associated with computing technologies that affect people's everyday activities and career options. [Practice 7.2 Communicating About Computing]

- The Proficient student:
  - Describes impacts associated with computing technologies that affect people's everyday activities.
  - Describes impacts associated with computing technologies that affect people's career options.

8.IC.SI.02 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [Practice 2.1 Collaborating Around Computing] [Practice 7.3 Communicating About Computing]

- The Proficient student:
  - Collaborates using tools to connect with peers when creating a computational artifact.
  - Practices grade-level appropriate behavior and responsibilities while participating in an online community.
  - Identifies and reports inappropriate behavior while participating in an online community, when applicable.

8.IC.SLE.02 Using grade level appropriate content and complexity, discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent. [Practice 1.1 Fostering an Inclusive Computing Culture] [Practice 7.2 Communicating About Computing]

- The Proficient student:
  - Describes tradeoffs between allowing information to be public and keeping information private and secure, and
  - With regard to positive and malicious intent:
    - » Discusses the legal impacts associated with software development and use,
    - » Discusses the social impacts associated with software development and use,
    - » Discusses the ethical impacts associated with software development and use.

9-12 Computer Science Content & Performance Standards

Introduction:
In high school, students will continue to develop their knowledge of computing systems, their components, and how systems interact. Students will use their understanding about the basic principles of computation, that algorithms describe a step-by-step solution to a problem, that programs are algorithms written in a language that a computer can understand, and that the solution to many problems can be described as a program. A solid foundation of algebraic concepts is important for success in high school computer science courses. Students will expand their ability to identify patterns and create algorithms that can model the observed patterns.

Computing Systems

Devices (D), Hardware & Software (HS), and Troubleshooting (T)

L1.CS.HS.01 Explain the interactions between application software, system software, and hardware layers. [Practice 4.1 Developing and Using Abstractions]

- The Proficient student:
  - Identifies the interactions between application software, system software, and hardware layers.
  - Defines the interactions between application software, system software, and hardware layers.
  - Explains the interactions between application software, system software, and hardware layers. For example, text editing software interacts with the operating system to receive input from the keyboard, convert the input to bits for storage, and interpret the bits as readable text to display on the monitor.

Network & The Internet

Network, Communication, & Organization (NCO) and Cybersecurity (C)

L1.NI.C.03 Compare various security measures, considering trade-offs between the usability and security of a computing system. [Practice 6.3 Testing and Refining Computational Artifacts]

- The Proficient student compares two or more security measures, considering trade-offs between the usability and security of a computing system.

Data Analysis

Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

L1.DA.CVT.01 Create interactive data representations using software tools to help others better understand real-world phenomena (e.g., paper surveys and online data sets). [Practice 4.4 Developing and Using Abstractions]

- The Proficient student creates appropriate interactive data representations using software tools to help others better understand real-world phenomena.

Algorithms & Programming

Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

L1.AP.A.01 Create a prototype that uses algorithms (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student. [Practice 5.2 Creating Computational Artifacts]

- The Proficient student creates a prototype that uses appropriate algorithms (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student.

L1.AP.A.02 Describe how artificial intelligence algorithms drive many software and physical systems. [Practice 7.2 Communicating About Computing]

- The Proficient student describes how artificial intelligence algorithms drive many software and physical systems.

L1.AP.C.03 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. [Practice 5.2 Creating Computational Artifacts]

- The Proficient student:
  - Designs and iteratively develops a computational artifact that uses events to initiate instructions for:
    - » Practical intent,
    - » Personal expression, or
    - » Societal issues.

L1.AP.M.01 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. [Practice 3.2 Recognizing and Defining Computational Problems]

- The Proficient student decomposes problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

L1.AP.PD.01 Plan and develop programs by analyzing a problem and/or process, developing and documenting a solution, testing outcomes, and adapting the program for a variety of users. [Practice 5.1 Creating Computational Artifacts]

- The Proficient student plans and develops programs by:
  - Analyzing a problem and/or process.
  - Developing and documenting a solution.
  - Testing outcomes.
  - Adapting the program for a variety of users.

L1.AP.PD.04 Design and develop computational artifacts, working in team roles, using collaborative tools. [Practice 2.4 Collaborating Around Computing]

- The Proficient student designs and develops computational artifacts while working in teams and using collaborative tools.

Impacts of Computing

Culture (C), Social Interactions (SI), and Safety, Law, and Ethics (SLE)

L1.IC.SLE.03 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. [Practice 7.3 Communicating About Computing]

- The Proficient student evaluates the following implications of privacy in the context of safety, law, or ethics:
  - Social implications
  - Economic implications

L1.IC.SLE.04 Using grade level appropriate content and complexity, discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent. [Practice 1.1 Fostering an Inclusive Computing Culture] [Practice 7.2 Communicating About Computing]

- The Proficient student, when considering both positive and malicious intent regarding software development and use, discusses the:
  - Legal impacts
  - Social impacts
  - Ethical impacts