# Computer Science Curriculum Guide 2021

ACADEMIC ACHIEVEMENT
LEARN
MONTANA OFFICE OF PUBLIC INSTRUCTION

# CONTENTS

# ACKNOWLEDGEMENTS

The following individuals contributed their time, expertise, and commitment to Montana's students through their participation in the standards adoption process.

Computer Science Writing Team

- Garth Flint
- Kerri Cobb
- Steve Qunell
- Kathi Hoyt
- Carla Olson

Computer Science Negotiated Rulemaking Committee

- Ann Ewbank
- Kar Sheppard
- Richard Gross
- Renee Rasmussen
- Gary Myers
- Shelly Stanton
- Shaundel Krumheuer
- Carla Swenson
- Noelle Harper
- Steve Qunell
- Melody Lee
- Mistyne Hall
- Nikolas Griffith
- Dee Hensley-Maclean
- Shannon Hanson

OPI Staff

- Eric Swenson
- Marisa Graybill

# INTRODUCTION

## WHAT IS A CURRICULUM GUIDE?

Montana Code Annotated TITLE 20. EDUCATION | CHAPTER 3. ELECTED OFFICIALS | Part 1. Superintendent of Public Instruction

> 20-3-106. Supervision of schools -- powers and duties. The superintendent of public instruction has the general supervision of the public schools and districts of the state and shall perform the following duties or acts in implementing and enforcing the provisions of this title: (19) collect and ***maintain a file of curriculum guides*** and assist schools with instructional programs in accordance with the provisions of 20-7-113 and 20-7-114;

This guide is designed to provide resources and guidance to schools, districts, curriculum consortia and others at multiple access points and stages of curriculum development. It allows educators to find pertinent information for a wide range of topics that inform the curriculum development process--from the legal foundations in Montana law and rules to classroom level instructional strategies and assessment. Use the topic outline to access information across the curriculum development spectrum.

This is a guide. It is not exhaustive in its depth or in the number of resources, but it is specifically designed for Montana educators to look with intention and clear guidance at improving the process of curriculum development in Montana schools.

## THEORY AND RESEARCH SUPPORTING A CURRICULUM GUIDE

Delineating the underpinnings for a theory of a model curriculum guide is a challenging task. First, the theory needs to refer to the giants—the writers and thinkers in the field whose work helps to define our task. The theory should explain how the structure of knowledge is related to a model for curriculum. It should also reference what research tells us about how people learn best. Finally, the theory should provide users of the curriculum guide with enough direction to help ensure that materials, tasks, and products exemplify the principles around which the theory has been developed—in a user-friendly format.

### Referring to the Giants

In looking to the leading thinkers in the field to find out what is known about the hallmarks of good curricula, these heroes surfaced: William James, Alfred North Whitehead, John Dewey, Hilda Taba, Ralph Tyler, Benjamin Bloom, Jerome Bruner, Carol Ann Tomlinson, Sandra Kaplan, Jann Leppien, Jean Purcell, H. Lynn Erickson, Grant Wiggins, Jay McTighe, Larry Lezotte, and Robert Marzano.

A synthesis of their thinking suggests that good curriculum should:

- Be organized around the structure of knowledge
- Reflect content selection and procedures (student tasks) that will help maximize the in-depth understanding and transfer of knowledge, understanding and skills
- Have a clear focus on the essential facts, understandings and skills that professionals in the disciplines value most (delineated in the state standards) and select content (representative topics) that best represent the essential structure of the disciplines
- Respect the unique characteristics of the learner
    - Recognize and support the need of each learner to make sense of ideas and information, reconstructing older understandings with new ones
    - Address interest and readiness levels
- Place a premium on the development of process skills (including skills of inquiry, thinking skills and technology integration skills) as well as the appropriate use of methodology within content fields
- Be aligned
    - all component parts with the goal of in-depth understanding (Tomlinson)

## STRUCTURE OF KNOWLEDGE

The content teachers and students wrestle within the classroom--history, science, mathematics, the study of language--all comes from the disciplines.

> “The disciplines have evolved as discrete entities over centuries as the result of the different kinds of questions researchers have asked and the different research methodologies they have developed to answer them.” (Renzulli)

Knowing that in order to design effective curriculum it is necessary to better understand how knowledge within a discipline is constructed, Hilda Taba, a powerful and insightful educator in the 1950s and 60s became a primary source. She advocated teaching to the deeper understanding of concepts and main ideas (transferrable, conceptual understandings) rather than focusing on superficial coverage of the factual information (Taba).

> Theories: Explanations of the nature or behavior of a specified set of phenomena based on the best evidence available. “The big bang theory of the universe.” “The land bridge of early human migration.”
>
> Principles: Two or more concepts stated in a relationship. Usually considered to be the foundational truths of a discipline. “The supply and demand of goods and services affect cost.” “Any straight line can be extended indefinitely in a straight line.”
>
> Generalizations: Two or more concepts stated as a relationship – essential learnings or understandings; the "big ideas" related to the critical concepts and topics of a study (e.g., “Organisms adapt to changing environments in order to survive.” “Numbers can be added together in different ways to reach a common sum.”)
>
> Concepts: One- or two-word concepts are abstract, timeless, transferable and universal. Concepts may be very broad macro-concepts, such as “change,” “system,” or “interdependence”; or they may be more topic specific, such as “organism,” “habitat,” or “culture.”
>
> Topics: The lens through which content is explored such as “Causes of the Revolution, “Rocks and Minerals,” or “Geometry.”
>
> Facts: Defined in the Montana Content Standards
>
> Skills: Defined in the Montana Content Standards (Erickson)

## HOW PEOPLE LEARN

In creating a conceptualization about how people learn best, sources such as *How People Learn* (Bransford)*, How Students Learn History, Science and Mathematics in the Classroom* (Donovan), curriculum models of Understanding by Design and the Parallel Curriculum Model provided research and insight. All of them support the notion that helping students organize their learning around big ideas and transferable concepts is essential.

The 2005 publication *How People Learn: Brain, Mind, Experience, and School* delves into scientific findings from studies of people who have developed expertise in a variety of areas. Of course, not all school children are expected to become experts, but the study of expertise does show what the results of successful learning look like. "The studies found that experts' knowledge is not simply a list of facts and formulas that are relevant to their domain; instead, their knowledge is organized around core concepts or "big ideas" that guide their thinking about their domains." (Bransford)

> "The idea that experts recognize features and patterns that are not noticed by novices is potentially important for improving instruction. Research on expertise suggests the importance of providing students with learning experiences that specifically enhance their abilities to recognize meaningful patterns of information." (Bransford)

> "The fact that experts' knowledge is organized around important ideas or concepts suggests that curricula should also be organized in ways that lead to conceptual understanding. Many approaches to curriculum design make it difficult for students to organize knowledge meaningfully. Often there is only superficial coverage of facts before moving on to the next topic; there is little time to develop important, organizing ideas." (Bransford)

To apply this important research to what actually happens in classrooms, the National Academies Press published *How Students Learn: History, Mathematics, and Science in the Classroom*. The authors make the following points:

1. Students come to the classroom with preconceptions about how the world works. If their initial understanding is not engaged, they may fail to grasp the new concepts and information, or they may learn them for purposes of a test but revert to their preconceptions outside the classroom.
2. To develop competence in an area of inquiry, students must (a) have a deep foundation of factual knowledge, (b) understand facts and ideas in the context of a conceptual framework, and (c) organize knowledge in ways that facilitate retrieval and application.

3. A “metacognitive” approach to instruction can help students learn to take control of their own learning by defining learning goals and monitoring their progress in achieving them. (Donovan)

## WHAT DO TEACHERS NEED FROM CURRICULUM GUIDES?

The last consideration in developing the theoretical underpinnings of the guide is that all component parts be aligned, lead to in-depth understanding and be responsive to teacher needs. From a September 2010 article in *Educational Leadership* “Among Colleagues: What do Teachers need from Curriculum Guides,” the following comments informed development of this guide:

> “Begin with the end in mind. What do I want students to understand when teachers have finished instruction?”
>
> “My advice is to err on the side of less. Give teachers a guide, not an ‘everything’ bagel.”
>
> “The staff development that supports the new curriculum is as important as the new curriculum itself.” (Among colleagues: what do teachers need from curriculum guides?)

The Third International Mathematics and Science Survey (TIMSS) criticized curricula that were "a mile wide and an inch deep" and argued that this is much more of a problem in America than in most other countries. Research on expertise suggests that a superficial coverage of many topics in the domain may be a poor way to help students develop the competencies that will prepare them for future learning and work. This guide to curriculum development seeks to help address this fundamental issue of American schools. (Wang)

# INDIAN EDUCATION FOR ALL INTEGRATION

Montana's constitutional requirement and duly enacted policy require recognition of the distinct and unique cultural heritage of American Indians and a commitment in our educational goals to preserve their cultural heritage. Every Montanan, whether Indian or non-Indian, should be encouraged to learn about the distinct and unique heritage of American Indians in a culturally responsive manner. The OPI Indian Education for All (IEFA) Unit works with districts, tribes, and other entities to ensure all schools have the knowledge, tools and resources necessary to honor the IEFA requirement and integrate it into their teaching materials and methods.

> Article X of the Montana Constitution
>
> Education and Public Lands (1972)
>
> Section 1
>
> (1) It is the goal of the people to establish a system of education which will develop the full educational potential of each person. Equality of educational opportunity is guaranteed to each person of the state.
>
> (2) The state recognizes the distinct and unique cultural heritage of the American Indians and is committed in its educational goals to the preservation of their cultural integrity.
>
> Indian Education for All (1999) MCA 20-1-501
>
> (2) It is the intent of the legislature that in accordance with Article X, section 1(2), of the Montana constitution:
>
> (a) every Montanan, whether Indian or non-Indian, be encouraged to learn about the distinct and unique heritage of American Indians in a culturally responsive manner; and
>
> (b) every educational agency and all educational personnel will work cooperatively with Montana tribes or those tribes that are in close proximity, when providing instruction………to include information specific to the cultural heritage and contemporary contributions of American Indians, with particular emphasis on Montana Indian tribal groups and governments.

(3) It is also the intent of this part, predicated on the belief that all school personnel should have an understanding and awareness of Indian tribes to help them relate effectively with Indian students and parents…

## IEFA INTEGRATION PROCESS

- Be sure you are familiar with the Essential Understandings Regarding Montana Indians (EUs).
- Find content area standards that specifically mention IEFA or could include an IEFA connection.
- Consider existing or new curricular areas where standards might be addressed.
- Identify and connect the appropriate EU(s).
- Select and identify any new curricular resources needed for solid IEFA integration.

## IDENTIFYING AND SELECTING IEFA CURRICULAR RESOURCES

- Check with your librarian to see what IEFA resources might be available. You can also review the list of IEFA resources that should be in your library.
- Visit the MT OPI Indian Education for All website for possible resources and ideas.
- Be sure that resources are culturally authentic and tribally specific.
- Utilize the MT OPI IEFA Publication: *Evaluating American Indian Materials and Resources for the Classroom* when necessary.
- Reach out to the MT OPI IEFA unit with any with any questions, ideas, or PD requests.

## IEFA FOUNDATIONAL DOCUMENTS

- Essential Understandings Regarding Montana Indians
    - Essential Understandings Key Concepts
    - Essential Understandings Poster
- Evaluating American Indian Materials and Resources for the Classroom
- The Framework: A Practical Guide for Montana Teachers and Administrators Implementing Indian Education for All
- Funding Spectrum Guide - Indian Education for All
- History and Foundation of American Indian Education
- Montana Indian Education for All Evaluation

# CURRICULUM DEVELOPMENT GUIDANCE AND RESOURCES

Montana teachers from all grade levels and content areas must include instruction that is aligned to state standards. It is up to local schools and teachers to design instructional strategies, select content to teach, and assess student progress toward those standards.

## ADMINISTRATIVE RULES OF MONTANA

10.55.603 CURRICULUM AND ASSESSMENT

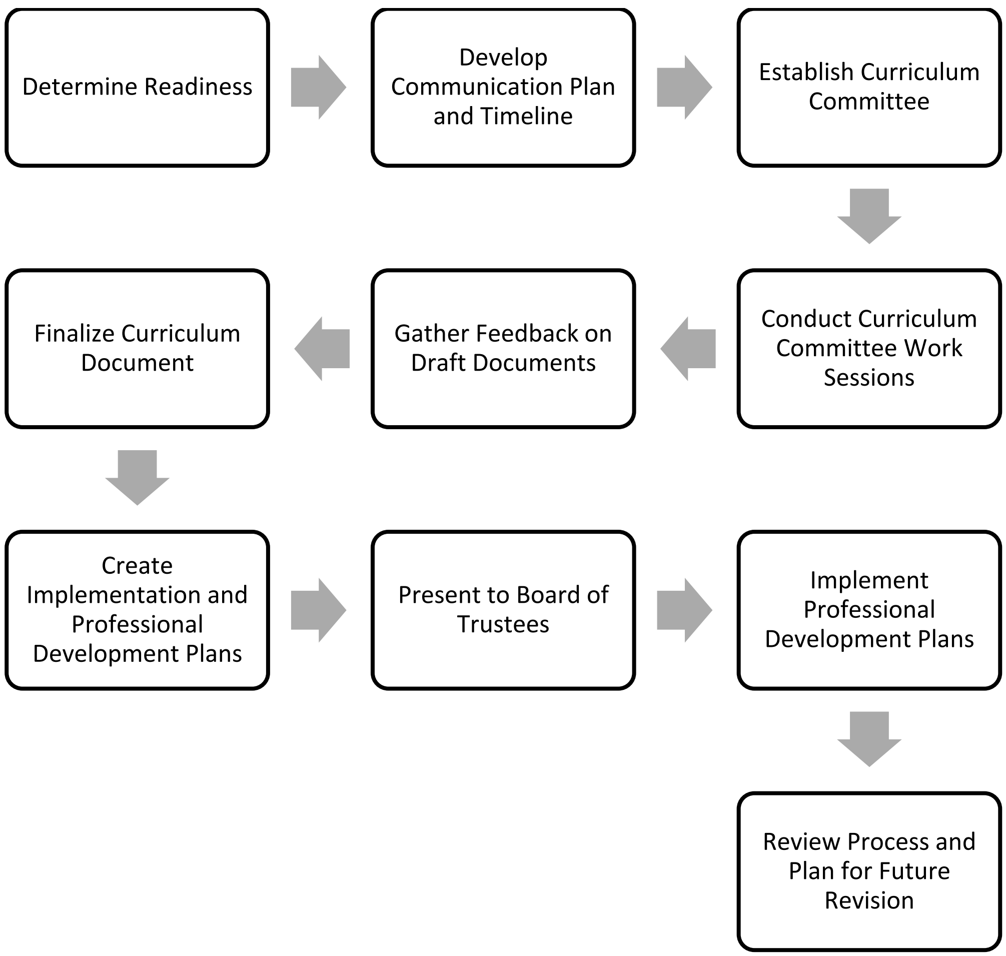
(1) Local school districts shall ensure their curriculum is aligned to all content standards and the appropriate learning progression for each grade level.

10.55.1001 PROGRAM STANDARDS

(1) It is the local board of trustees' responsibility to ensure the district's curricula align with the state content standards and content-specific grade-level learning progressions.

## CURRICULUM DEVELOPMENT PHASES

Curriculum development is a continuous process. The phases described in this chart are a suggested progression. An emphasis on examination of resources, stakeholder involvement, professional feedback, and implementation planning is essential to ensure a successful outcome for curriculum development and revision.



## Readiness Checklist for Curriculum Development

| Outcomes | Level of Readiness | Notes/Dates |
|---|---|---|
| The content standards and learning expectations for each grade and subject area have been identified and are known to the teachers expected to teach them. | ○ Have Not Begun<br>○ Just Underway<br>○ Making Good Progress<br>○ Successfully Completed<br>○ Prepared to Share with Others | |
| Teachers have "mapped" their grade level and/or subject matter standards onto an annual school calendar thus producing instructional "pacing charts" where critical milestones and benchmarks are known. | ○ Have Not Begun<br>○ Just Underway<br>○ Making Good Progress<br>○ Successfully Completed<br>○ Prepared to Share with Others | |
| Student assessments have been identified or developed and aligned with the standards. | ○ Have Not Begun<br>○ Just Underway<br>○ Making Good Progress<br>○ Successfully Completed<br>○ Prepared to Share with Others | |
| Individual teachers use the pacing charts and formative assessments to plan and deliver classroom instruction. | ○ Have Not Begun<br>○ Just Underway<br>○ Making Good Progress<br>○ Successfully Completed<br>○ Prepared to Share with Others | |
| Throughout the school year teachers engage in horizontal (e.g., grade level) and vertical (e.g., cross-grade level) conversations to be sure that every student has an opportunity to master ALL learning expectations required for student success at the next grade level. | ○ Have Not Begun<br>○ Just Underway<br>○ Making Good Progress<br>○ Successfully Completed<br>○ Prepared to Share with Others | |

(Lezotte)

## CURRICULUM DEVELOPMENT STAKEHOLDERS AND PARTICIPANTS

Consider the list below when selecting members for curriculum development focus groups and committees. The curriculum committee should be inclusive of all stakeholders. The committee chair should be determined by the district leadership team. Determine which stakeholders will participate in the full process and those who will help to provide feedback and expert review during the development cycle.

| Stakeholder Group | Suggested Roles | Expertise |
|---|---|---|
| District Leadership<br>• superintendent<br>• curriculum director<br>• building principal(s) | committee chair<br>committee member<br>focus group facilitator | • manage process<br>• provide resources to support process |
| Trustee Representative(s) | committee member | • link to Board of Trustees<br>• link to community |
| Teachers<br>• primary grades<br>• middle grades<br>• high school | committee chair<br>committee member<br>focus group participant | • content specialized teachers verify content<br>• content specialized teachers verify learning progressions K-12<br>• provide assessment expertise |
| Teacher-librarians | committee member<br>focus group participant | • provide insight into resources and instructional practice<br>• provide expertise in the inclusion of information and technology literacy in all curricula |

| Stakeholder Group | Suggested Roles | Expertise |
| --- | --- | --- |
| Specialists<br>• instructional coaches<br>• gifted and talented<br>• technology<br>• special education<br>• career and technical education<br>• fine and performing arts | committee member<br>focus group participant | • provide cross-curricular representation<br>• provide cross-grade expertise<br>• provide instructional expertise |
| Indian Education for All (IEFA) Specialists | committee member<br>focus group participant | • link to resources and best practices for implementing IEFA in curriculum |
| Guidance Counselors | focus group participant | • link to career development<br>• link to students |
| Parent Representative(s) | committee member<br>focus group participant | • link to community<br>• link to local concerns and issues |
| Student Representative(s) | committee member<br>focus group participant | • link to learners<br>• help check for relevance and engagement |
| Business and Community Representative(s) | focus group participant | • link to community<br>• link to local concerns and issues |

## CURRICULUM DEVELOPMENT WORK SESSION GOALS

The work of the curriculum committee can be organized and framed in each of the work sessions described below. This process and suggested goals should be adapted, and the number of sessions should be adjusted according to the scope of the revision or re-writing of the curriculum documents. An emphasis on examination of resources, public comment, professional feedback and implementation planning is essential to ensure a successful outcome for curriculum development and revision.

| | |
|---|---|
| Work Session 1 | •Establish Roles<br>•Review Montana program delivery standards<br>•Review Montana content standards<br>•Review local curriculum documents<br>•Review curriculum consortia documents (if applicable)<br>•Review other Montana district curriculum documents |
| Work Session 2 | •Revision and writing of curriculum documents<br>•Compare local curriculum to Montana content standards<br>•Identify gaps in existing curriculum documents |
| Work Session 3 | •Revision and writing of curriculum documents<br>•Examine learning progressions<br>•Gather feedback from content area teachers<br>•Gather feedback from community members, parents and students |
| Work Session 4 | •Revision and writing of curriculum documents<br>•Review feedback from content area teachers<br>•Review feedback from community members, parents and students<br>•Analyze for inclusion of Indian Education for All Resources and Seven Essential Understandings Regarding Montana Indians |
| Work Session 5 | •Revision and writing of curriculum documents<br>•Match resources to curriculum<br>•Identify additional resources needed<br>•Create implementation plan<br>•Create professional development plan |
| Work Session 6 | •Review Goals<br>•Check Vertical and Horizontal Alignment<br>•Check learning progressions<br>•Prepare final document(s)<br>•Prepare for public comment and feedback<br>•Prepare for presentation to Board of Trustees |

# PROGRAM DELIVERY PLANNING

## ADMINISTRATIVE RULES OF MONTANA

### 10.55.1003 PROGRAM FOUNDATION STANDARDS

(1) The purpose of all programs is to develop and apply knowledge and skills necessary to pursue lifelong goals and opportunities.

(2) Program foundation standards are the common conditions and practices that will be evident in all programs within a school system to ensure that all students have educational opportunity to learn, develop, and demonstrate learning in the content standards and content-specific grade-level learning progressions. All programs shall follow the content standards in the accreditation rules of Montana. The local board of trustees shall:

- (a) meet the following conditions:
  - (i) ensure integration of the history, contemporary portrayals, and contributions of American Indians, with an emphasis on Montana Indians, for all students, across all content areas;
  - (ii) ensure an educational climate that promotes academic freedom and respect for diversity with prejudice toward none;
  - (iii) maintain high expectations for student performance, behavior, and lifelong learning; and
  - (iv) encourage collaboration among school personnel to plan, assess, and support instruction.
- (b) include the following practices:
  - (i) offer engaging and relevant experiences that enable students to develop effective communication skills in their personal lives, workplaces, and communities;
  - (ii) teach ethical behavior, including use of technology (social media) and the implications of one's choices;
  - (iii) implement research-based instructional skills and strategies to improve student learning;
  - (iv) challenge students to think creatively and critically, and use the inquiry process to solve problems and make informed decisions;
  - (v) encourage interdisciplinary instruction;
  - (vi) use relevant data to inform decision making, modify instruction, and increase student learning;

(vii) integrate information literacy skills, technology tools, and workplace competencies to support learning in all curricular areas; and
(viii) provide equitable access to all facilities, technology, equipment, materials, and services necessary to support the instructional process.

History: 20-2-114, MCA; IMP, 20-2-121, 20-3-106, 20-7-101, MCA; NEW, 2001 MAR p. 953, Eff. 6/8/01; AMD, 2003 MAR p. 554, Eff. 3/28/03; AMD, 2006 MAR p. 755, Eff. 3/24/06; AMD, 2012 MAR p. 2042, Eff. 7/1/13.

## PROGRAM-SPECIFIC DELIVERY STANDARDS

### COMPUTER SCIENCE PROGRAM DELIVERY STANDARDS 10.55.2201

In general, a basic program in computer science education shall:

- meet the following conditions:
    - provide a well-articulated integrated curriculum that challenges students to learn increasingly more sophisticated computer science concepts across all grade levels and content areas wherever appropriate; and
    - foster a collaborative environment that embraces creativity, communication, and problem solving;
- include the following practices:
    - ensure students become informed citizens who can critically engage in public discussion on computer science related topics;
    - ensure students develop as learners, users, and creators of computer science knowledge and artifacts;
    - ensure students understand the role and impact of computing in the world around them, leveraging computer technology to create solutions; and
    - increase career and college readiness.

## COMPUTER SCIENCE LICENSURE REQUIREMENTS

All elementary and middle school certified teachers can teach computer science.

An individual may teach Computer Science courses at the secondary level via:

- Class 1 or 2 license with either the Business Education 5-12 (Broadfield) [12S BUS], Computer & Information Sciences 5-12 [12S CS], or Computer Information Systems endorsement [12S CIS] (all courses*).
- Class 1 or 2 license with a Mathematics 5-12 endorsement [02S MAT] (only two courses*).
- Class 1 or 2 license, any endorsement, with the addition of a Class 4 Computer Coding endorsement [Class 1 or 2 license + 10S CC] (only computer coding/computer programming courses*).**
- Class 4 Career & Technical Education license with the Computer Information Systems endorsement [10S CIS] (all courses*).

*Please refer to the OPI Course Codes for a list of courses that can be taught by the various endorsements above. Computer Science related courses are listed under the 010-Computer and Information Sciences and 011-Communications and Audio/Visual Technology sections.

# STANDARDS

## OVERVIEW

The Montana Content Standards for Computer Science (implemented July 1, 2021) are based off of the Computer Science Teachers Association (CSTA) K-12 Computer Science Standards. The Montana Content Standards for Computer Science provide the foundation for a complete computer science program from kindergarten through high school. The Standards are written for grades K, 1, 2, 3, 4, 5 and grade banded for 6-8 and 9-12. The Standards are written for *all* students to engage in with the following end goals:

- Introduce the fundamental concepts of computer science to all students, beginning at the elementary school level.
- Present computer science at the secondary school level in a way that can fulfill a math graduation credit.
- Encourage schools to offer additional secondary-level computer science courses that will allow interested students to study facets of computer science in more depth and prepare them for entry into the workforce or college.
- Increase the availability of rigorous computer science for all students, especially those who are members of underrepresented groups.

*Source: Computer Science Teachers Association (2017). CSTA K-12 Computer Science Standards, Revised 2017. Retrieved from http://www.csteachers.org/standards.*

## WHAT EDUCATORS NEED TO KNOW ABOUT THE NEW STANDARDS/WHAT'S DIFFERENT?

The Montana Content Standards for Computer Science are organized by content area and skills. The content areas, or core concepts, are categories representing major areas in the field of computer science. The content areas represent specific areas of disciplinary importance rather than abstract, general ideas. The computer science content areas for the Standards are listed below. Each grade level contains all or some of the core concept areas.

The computer science content areas of the Standards are:

1. Algorithms and Programming
2. Computing Systems
3. Data and Analysis

4. Impacts of Computing
5. Computer Science Networks and the Internet

The Standards also include a set of seven skills, or practices, which are the behaviors computationally literate students can use to fully engage with the core concepts of computer science.

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

Computational thinking is at the heart of the computer science practices and is delineated by skills 3–6. Skills 1, 2, and 7 are independent, general practices in computer science that complement computational thinking. The skills are not delineated by grade. Rather, the skills use a narrative to describe how students should exhibit each skill with increasing sophistication from kindergarten to Grade 12.

The content areas and skills are integrated to provide experiences for computer science students K-12.

## K-12 RESOURCES FOR IMPLEMENTATION

**Example K–12 Pathways**

The Montana Content Standards outline a K–12 progression in computer science that builds sophistication over time. Course and instructional pathways that treat computer science as a discipline, as opposed to an individual elective, are required to implement this vision. Courses and curriculum do not exist in an instructional vacuum and should take into account the content areas and skills as laid out across grades. This section explores options and resources for building a K–12 instructional pathway in computer science. Below are experts from the [CSTA framework] that can be used to help in implementation of a K-12 Computer Science pathway.

In the example pathways in Figure 1, a computer science experience can range from a few hours a week to a semester- or year-long course. The different models in each grade band are organized by an estimate of the total amount of focused computer science instructional time that the model may allow, from least to greatest. The examples within each grade band are not mutually exclusive; many options can be combined to create additional avenues for computer science instruction.

Elementary Implementation Examples

- Integrated into general curriculum
- Integrated into an existing exploratory course (i.e., media arts, computer lab)
- Create an independent course, similar to Science, Math, or Music

Middle School Implementation Examples

- Integrated into business/computer/keyboarding courses or within other subjects, such as industrial technology, math, science, or other subjects
- Create an independent course at a particular grade level or at all grade levels

High School Implementation Examples

- Integrated into business, industrial technology, math, science, or other subjects
- Create an independent course
- Offer as an independent study course
- Instruction to include introductory, advanced (AP), and specialized courses (i.e, app development, cybersecurity, game design, networking)

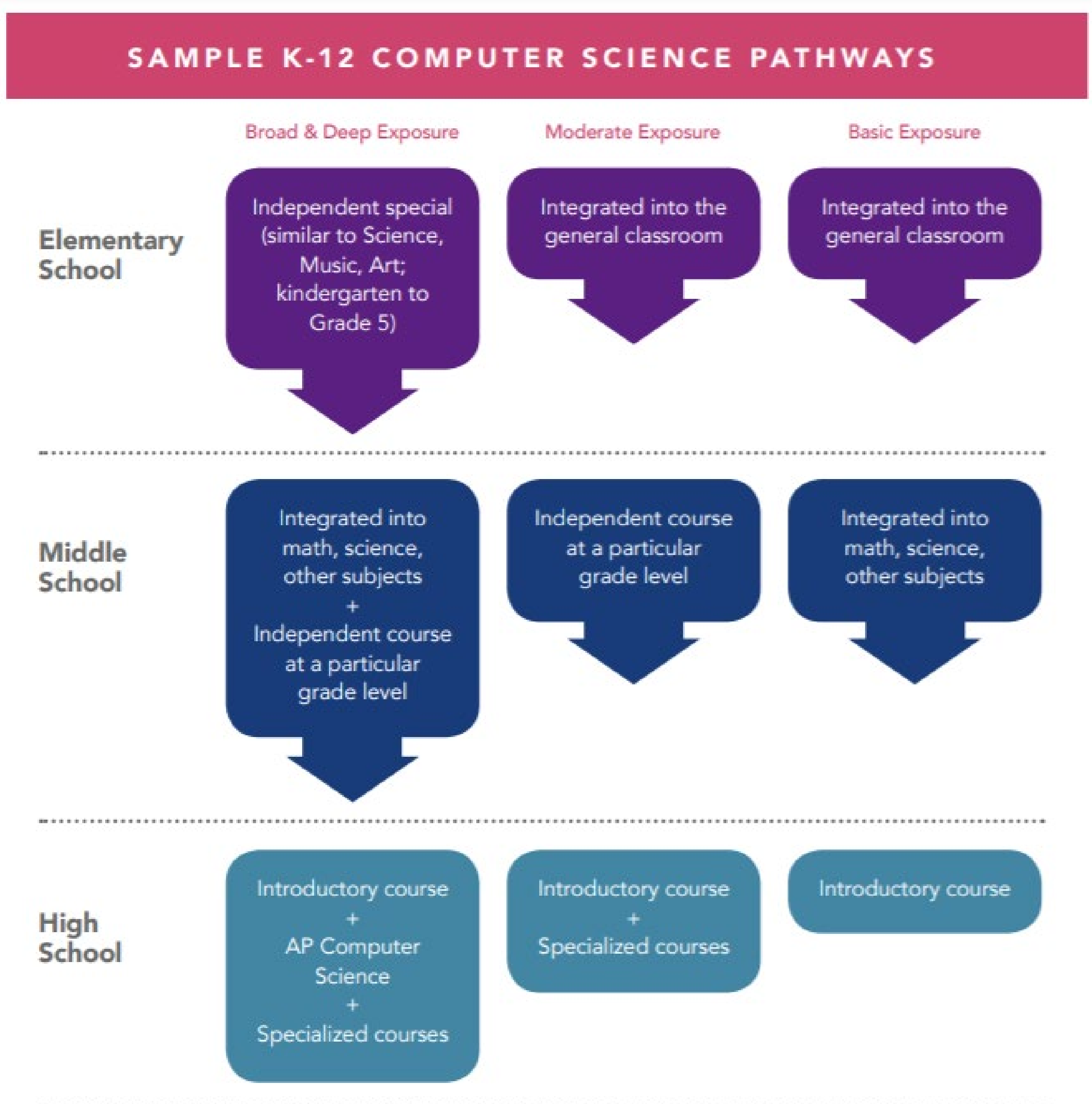



**FIGURE 1: K-12 COMPUTER SCIENCE PATHWAYS**

**Technical Infrastructure**
Modern computer science education does not often require complex hardware setups or software configurations, although some basic technical conditions must be in place. The availability of computers is essential. Some common strategies at the classroom level for dealing with a lack of computers are having students work collaboratively on devices, having a center for computing (particularly at the elementary level), or rotating students through a computing experience within a weekly schedule.

While some computer science education curricula and programs require locally installed resources, many computer science education programs and curricula now exist online. Using programs and curricula that are online allows schools to cover much of the curriculum with browser-capable devices, including netbooks and, in some cases, tablets. Using a web-based solution to support computer science curricula does require a robust network connection. Schools will want to weigh their options based upon their own current technological resources.

**Stakeholders**
While physical infrastructure is important, equally important is cultivation of community stakeholders who can support computer science implementation. Just like other subject areas, all students are capable of learning the basics of computer science, and it is widely agreed in the computer science education community that the fundamentals of computer science are essential for developing critical thinking skills and understanding the technology that people interact with daily.

**Instructional and Professional Learning Resources**
Computer Science is a growing field. There are hundreds of resources for curriculum and professional development available. A curated list is available in Appendix A and on the OPI Computer Science webpage (http://opi.mt.gov/Educators/Teaching-Learning/K-12-Standards-Revision/Computer-Science).

## Ideas for Integration

Integration is an essential component for successful K-12 implementation of the Computer Science Standards. Refer to Figure 1 for examples of how an integrated pathway may look. Appendix A contains a series of curated resources. Many of the recourses are created with the intention of integration into other subjects areas.

## Content Areas and Instructional Units Overview

Below is an in-depth description of the core content areas, or concepts. The core content areas are delineated by multiple subconcepts that represent specific ideas within each concept.

**Algorithms and Programming:** 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. Algorithms and programming control 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.

- **Algorithms:** Algorithms are designed to be carried out by both humans and computers. 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, as well as the evaluation of competing algorithms.

- **Variables:** Computer programs store and manipulate data using variables. In early grades, students learn that different types of data, such as words, numbers, or pictures, can be used in different ways. As they progress, students learn about variables and ways to organize large collections of data into data structures of increasing complexity.

- **Control:** Control structures specify the order in which instructions are executed within an algorithm or program. In early grades, students learn about sequential execution and simple control structures. As they progress, students expand their understanding to combinations of structures that support complex execution.

- **Modularity:** Modularity involves breaking down tasks into simpler tasks and combining simple tasks to create something more complex. In early grades, students learn that algorithms and programs can be designed by breaking tasks into smaller parts and recombining existing solutions. As they progress, students learn about recognizing patterns to make use of general, reusable solutions for commonly occurring scenarios and clearly describing tasks in ways that are widely usable.

- **Program Development:** Programs are developed through a design process that is often repeated until the programmer is satisfied with the solution. In early grades, students learn how and why people develop programs. As they progress, students learn about the tradeoffs in program design associated with complex decisions involving user constraints, efficiency, ethics, and testing.

**Computing Systems:** People interact with a wide variety of computing devices that collect, store, analyze, and act upon information in ways that can affect human capabilities both positively and negatively. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. An understanding of hardware and software is useful when troubleshooting a computing system that does not work as intended.

- **Devices:** Many everyday objects contain computational components that sense and act on the world. In early grades, students learn features and applications of common computing devices. As they progress, students learn about connected systems and how the interaction between humans and devices influences design decisions.

- **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 interaction between hardware and software at multiple levels within computing systems.

- **Troubleshooting:** When computing systems do not work as intended, troubleshooting strategies help people solve the problem. In early grades, students learn that identifying the problem is the first step to fixing it. As they progress, students learn systematic problem-solving processes and how to develop their own troubleshooting strategies based on a deeper understanding of how computing systems work.

**Data and Analysis:** Computing systems exist to process data. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important. Data is collected and stored so that it can be analyzed to better understand the world and make more accurate predictions.

- **Collection:** Data is collected with both computational and noncomputational tools and processes. In early grades, students learn how data about themselves and their world is collected and used. As they progress, students learn the effects of collecting data with computational and automated tools.

- **Storage:** Core functions of computers are storing, representing, and retrieving data. In early grades, students learn how data is stored on computers. As they progress, students learn how to evaluate different storage methods, including the tradeoffs associated with those methods.

- **Visualization and Transformation:** Data is transformed throughout the process of collection, digital representation, and analysis. In early grades, students learn how transformations can be used to simplify data. As they progress, students learn about more complex operations to discover patterns and trends and communicate them to others.

- **Inference and Models:** Data science is one example where computer science serves many fields. Computer science and science use data to make inferences, theories, or predictions based upon the data collected from users or simulations. In early grades, students learn about the use of data to make simple predictions. As they progress, students learn how models and simulations can be used to examine theories and understand systems and how predictions and inferences are affected by more complex and larger data sets.

**Impacts of Computing:** Computing affects many aspects of the world in both positive and negative ways at local, national, and global levels. Individuals and communities influence computing through their behaviors and cultural and social interactions, and in turn, computing influences new cultural practices. An informed and responsible person should understand the social implications of the digital world, including equity and access to computing.

- **Culture:** Computing influences culture—including belief systems, language, relationships, technology, and institutions—and culture shapes how people engage with and access computing. In early grades, students learn how computing can be helpful and harmful. As they progress, students learn about tradeoffs associated with computing and potential future impacts of computing on global societies.

- **Social Interactions:** Computing can support new ways of connecting people, communicating information, and expressing ideas. In early grades, students learn that computing can connect people and support interpersonal communication. As they progress, students learn how the social nature of computing affects institutions and careers in various sectors.

- **Safety, Law, and Ethics:** Legal and ethical considerations of using computing devices influence behaviors that can affect the safety and security of individuals. In early grades, students learn the fundamentals of digital citizenship and appropriate use of digital media. As they progress, students learn about the legal and ethical issues that shape computing practices.

**Computer Science Networks and the Internet:** Computing devices typically do not operate in isolation. Networks connect computing devices to share information and resources and are an increasingly integral part of computing. Networks and communication systems provide greater connectivity in the computing world by providing fast, secure communication and facilitating innovation.

- **Network Communication and Organization:** Computing devices communicate with each other 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.

- **Cybersecurity:** Transmitting information securely across networks requires appropriate protection. In early grades, students learn how to protect their personal information. As they progress, students learn increasingly complex ways to protect information sent across networks.

## Computer Science Skills Overview

When a district incorporates or integrates computer science content into district curriculum or offers a course in computer science, the following skills at each grade level apply:

**Fostering an Inclusive Computing Culture:** Building an inclusive and diverse computing culture requires strategies for incorporating perspectives from people of different genders, ethnicities, and abilities. Incorporating these perspectives involves understanding the personal, ethical, social, economic, and cultural contexts in which people operate. Considering the needs of diverse users during the design process is essential to producing inclusive computational products.

**Collaborating Around Computing:** Collaborative computing is the process of performing a computational task by working in pairs and on teams. Because it involves asking for the contributions and feedback of others, effective collaboration can lead to better outcomes than working independently. Collaboration requires individuals to navigate and incorporate diverse perspectives, conflicting ideas, disparate skills, and distinct personalities. Students should use collaborative tools to effectively work together and to create complex artifacts.

**Recognizing and Defining Computational Problems:** The ability to recognize appropriate and worthwhile opportunities to apply computation is a skill that develops over time and is central to computing. Solving a problem with a computational approach requires defining the problem, breaking it down into parts, and evaluating each part to determine whether a computational solution is appropriate.

**Developing and Using Abstractions:** Abstractions are formed by identifying patterns and extracting common features from specific examples to create generalizations. Using generalized solutions and parts of solutions designed for broad reuse simplifies the development process by managing complexity.

**Creating Computational Artifacts:** The process of developing computational artifacts embraces both creative expression and the exploration of ideas to create prototypes and solve computational problems. Students create artifacts that are personally relevant or beneficial to their community and beyond. Computational artifacts can be created by combining and modifying existing artifacts or by developing new artifacts. Examples of computational artifacts include programs, simulations, visualizations, digital animations, robotic systems, and apps.

**Testing and Refining Computational Artifacts:** Testing and refinement is the deliberate and iterative process of improving a computational artifact. This process includes debugging (identifying and fixing errors) and comparing actual outcomes to intended outcomes. Students also respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts.

**Communicating About Computing:** Communication involves personal expression and exchanging ideas with others. In computer science, students communicate with diverse audiences about the use and effects of computation and the appropriateness of computational choices. Students write clear comments, document their work, and communicate their ideas through multiple forms of media. Clear communication includes using precise language and carefully considering possible audiences.

# CODING THE STANDARDS

## GENERAL CODING SCHEME

SUBJECT.STANDARD.GradeLevel SUBSTANDARD.sub-substandard

CS = computer science (SUBJECT)
AP = algorithms and programming
CS = computing systems
DA = data and analysis
IC = impacts of computing
NI = computer science networks and the internet

Example:

| | |
|---|---|
| CS.DA.K | Computer science data and analysis standards for kindergarten are that each student will |
| CS.DA.K.1 | collect and categorize data |

***Note on identifying subjects and sub-content areas***

As a rule, when a content area has sub-content areas, (i.e. Civics and Government within Social Studies or Algorithms and Programming within Computer Science) the content area abbreviation is listed first before grade level and the sub-content area is listed second before grade level. For example: SS.CG.K is the coding for Social Studies> Civics and Government> Kindergarten. When a content area does not have specifically designated sub-content areas, such as in Career and Technical Education and Library Media, then the content area is used solely to identify the subject. For example: CTE.K is the human code for Career and Technical Education in Kindergarten.

## K-12 CONTENT AND SKILLS STANDARDS

The following standards will go into effect on July 1, 2021.

| Content | The content areas covered by the computer science standards may include |
|---|---|
| **AP** | algorithms and programming |
| **CS** | computing systems |
| **DA** | data and analysis |
| **IC** | impacts of computing |
| **NI** | computer science networks and the internet |

| Skills | When a district incorporates or integrates computer science content into district curriculum or offers a course in computer science, the following skills at each grade level apply |
|---|---|
| **CS.1.K12** | fostering an inclusive computing culture |
| **CS.2.K12** | collaborating around computing |
| **CS.3.K12** | recognizing and defining computational problems |
| **CS.4.K12** | developing and using abstractions |
| **CS.5.K12** | creating computational artifacts |
| **CS.6.K12** | testing and refining computational artifacts |
| **CS.7.K12** | communicating about computing |

# GRADE LEVEL STANDARDS

## KINDERGARTEN

### STANDARDS

| CS.K | COMPUTER SCIENCE CONTENT STANDARDS FOR KINDERGARTEN |
|---|---|
| **CS.AP.K** | **Computer science algorithms and programming standards for kindergarten are that each student will** |
| **CS.AP.K.1** | follow step-by-step instructions |
| **CS.AP.K.2** | recognize that numbers and symbols represent information |
| **CS.CS.K** | **Computer science computing systems standards for kindergarten are that each student will** |
| **CS.CS.K.1** | identify computing devices |
| **CS.CS.K.2** | identify examples of common hardware and software |
| **CS.DA.K** | **Computer science data and analysis standards to kindergarten are that each student will** |
| **CS.DA.K.1** | collect and categorize data |
| **CS.DA.K.2** | retrieve information |
| **CS.DA.K.3** | identify patterns in data |
| **CS.IC.K** | **Computer science impacts of computing standards for kindergarten are that each student will** |
| **CS.IC.K.1** | work respectfully and responsibly in groups |
| **CS.IC.K.2** | keep login information private and log off devices appropriately |

## FIRST GRADE

### STANDARDS

| **CS.1** | **COMPUTER SCIENCE CONTENT STANDARDS FOR FIRST GRADE** |
|---|---|
| **CS.AP.1** | **Computer science algorithms and programming standards for first grade are that each student will** |
| **CS.AP.1.1** | retell step-by-step instructions to complete a task |
| **CS.AP.1.2** | use numbers and symbols to represent information |
| **CS.AP.1.3** | arrange sequences and simple loops in correct order |
| **CS.CS.1** | **Computer science computing systems standards for first grade are that each student will** |
| **CS.CS.1.1** | identify tasks that can be performed by computing devices |
| **CS.CS.1.2** | use appropriate terminology in identifying common hardware and software problems |
| **CS.CS.1.3** | identify simple hardware and software problems |
| **CS.DA.1** | **Computer science data and analysis standards for first grade are that each student will** |
| **CS.DA.1.1** | collect and categorize data in up to three categories |
| **CS.DA.1.2** | retrieve, arrange and modify information |
| **CS.DA.1.3** | identify patterns in data |
| **CS.IC.1** | **Computer science impacts of computing standards for first grade are that each student will** |
| **CS.IC.1.1** | work respectfully and responsibly in groups |
| **CS.IC.1.2** | keep login information private and log off devices appropriately |

## SECOND GRADE

### STANDARDS

| CS.2 | COMPUTER SCIENCE CONTENT STANDARDS FOR SECOND GRADE |
|---|---|
| **CS.AP.2** | **The computer science algorithms and programming standards for second grade are that each student will** |
| **CS.AP.2.1** | model daily processes by creating and following sets of step-by-step instructions to complete tasks |
| **CS.AP.2.2** | model the way programs store and manipulate data by using numbers or other symbols to represent information |
| **CS.AP.2.3** | develop programs with sequences and simple loops to express ideas or address a problem |
| **CS.AP.2.4** | break down the steps needed to solve a problem into a precise sequence of instructions |
| **CS.CS.2** | **Computer science computing systems standards for second grade are that each student will** |
| **CS.CS.2.1** | select and operate appropriate tools to perform a variety of tasks |
| **CS.CS.2.2** | use appropriate terminology in identifying and describing the function of common hardware and software |
| **CS.CS.2.3** | describe basic hardware and software problems using accurate terminology |
| **CS.DA.2** | **Computer science data and analysis standards for second grade are that each student will** |
| **CS.DA.2.1** | collect and present the data in various visual formats |
| **CS.DA.2.2** | define data as gathered and stored information |
| **CS.DA.2.3** | identify and describe patterns in data visualizations, such as charts or graphs, to make predictions |
| **CS.IC.2** | **Computer science impacts of computing standards for second grade are that each student will** |
| **CS.IC.2.1** | identify how computing technology has changed how people live and work |
| **CS.IC.2.2** | work respectfully and responsibly online |
| **CS.IC.2.3** | keep login information private and log off devices appropriately |
| **CS.NI.2** | **Computer science networks and the internet standards for second grade are that each student will** |
| **CS.NI.2.1** | explain what passwords are and why they are used |
| **CS.NI.2.2** | recognize that computing devices and the internet enable people to connect with other people, places, information, and ideas |

## THIRD GRADE

### STANDARDS

| CS.3 | COMPUTER SCIENCE CONTENT STANDARDS FOR THIRD GRADE |
|---|---|
| **CS.AP.3** | **Computer science algorithms and programming standards for third grade are that each student will** |
| **CS.AP.3.1** | compare and contrast multiple algorithms to complete the same task |
| **CS.AP.3.2** | break down problems into smaller, manageable subproblems to facilitate the program development process |
| **CS.AP.3.3** | describe steps taken and choices made during the process of program development |
| **CS.AP.3.4** | identify intellectual property rights and give appropriate credit when creating or remixing programs |
| **CS.CS.3** | **Computer science computing systems standards for third grade are that each student will** |
| **CS.CS.3.1** | identify the internal and external parts of computing devices |
| **CS.CS.3.2** | determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies |
| **CS.DA.3** | **Computer science data and analysis standards for the third grade are that each student will** |
| **CS.DA.3.1** | collect data from multiple sources and display the data in graphs |
| **CS.DA.3.2** | describe multiple types of data |
| **CS.DA.3.3** | understand the accuracy of predictions and how they are influenced by the amount of data collected |
| **CS.IC.3** | **Computer science impacts of computing standards for third grade are that each student will** |
| **CS.IC.3.1** | collect diverse perspectives for the purpose of improving computational artifacts |
| **CS.IC.3.2** | identify rules associated with the appropriate use of digital information when creating computational artifacts |

| CS.3 | COMPUTER SCIENCE CONTENT STANDARDS FOR THIRD GRADE |
|---|---|
| **CS.IC.3.3** | describe ethical issues that relate to computing devices and networks |
| **CS.NI.3** | **Computer science networks and the internet standards for third grade are that each student will identify how personal information can be protected** |

## FOURTH GRADE

### STANDARDS

| CS.4 | COMPUTER SCIENCE CONTENT STANDARDS FOR FOURTH GRADE |
|---|---|
| **CS.AP.4** | **Computer science algorithms and programming standards for fourth grade are that each student will** |
| **CS.AP.4.1** | compare and refine multiple algorithms for the same task and determine which is the most appropriate |
| **CS.AP.4.2** | break down problems into smaller, manageable subproblems to facilitate the program development process |
| **CS.AP.4.3** | test and debug a program or algorithm to ensure it runs as intended |
| **CS.CS.4** | **Computer science computing systems standards for fourth grade are that each student will** |
| **CS.CS.4.1** | explain the function of individual internal and external parts |
| **CS.CS.4.2** | determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies |
| **CS.DA.4** | **Computer science data and analysis standards for fourth grade are that each student will** |
| **CS.DA.4.1** | select and use appropriate non-digital and digital tools to collect and represent data |
| **CS.DA.4.2** | identify and use multiple types of data to complete a task |
| **CS.DA.4.3** | evaluate the validity of data based on accuracy and relevance |
| **CS.IC.4** | **Computer science impacts of computing standards for fourth grade are that each student will** |
| **CS.IC.4.1** | collect diverse perspectives for the purpose of improving computational artifacts |
| **CS.IC.4.2** | identify rules associated with the appropriate use of digital information when creating computational artifacts |
| **CS.IC.4.3** | describe ethical issues that relate to computing devices and networks. |
| **CS.NI.4** | **Computer science networks and the internet standards for fourth grade are that each student will identify cybersecurity problems** |

# FIFTH GRADE

## STANDARDS

| CS.5 | COMPUTER SCIENCE CONTENT STANDARDS FOR FIFTH GRADE |
|---|---|
| **CS.AP.5** | **Computer science algorithms and programming standards for fifth grade are that each student will** |
| **CS.AP.5.1** | compare and refine multiple algorithms for the same task and determine which is the most appropriate |
| **CS.AP.5.2** | create programs that use variables to store and modify data |
| **CS.AP.5.3** | create programs that include sequences, events, loops, and conditionals |
| **CS.AP.5.4** | modify, remix, or incorporate portions of an existing program to develop something new or add more advanced features |
| **CS.AP.5.5** | describe choices made during program development |
| **CS.CS.5** | **Computer science computing systems standards for fifth grade are that each student will** |
| **CS.CS.5.1** | describe how internal and external parts of computing devices function to form a system |
| **CS.CS.5.2** | model how computer hardware and software work together as a system to accomplish tasks |
| **CS.CS.5.3** | determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies |
| **CS.DA.5** | **Computer science data and analysis standards for fifth grade are that each student will** |
| **CS.DA.5.1** | organize and present collected data visually to highlight relationships and support a claim |
| **CS.DA.5.2** | demonstrate how to store, copy, search, retrieve, modify, and delete information using a computing device |
| **CS.DA.5.3** | use accurate and relevant data to highlight or propose cause-and-effect relationships, predict outcomes, or communicate an idea |
| **CS.IC.5** | **Computer science impacts of computing standards for fifth grade are that each student will** |
| **CS.IC.5.1** | explain how computing technologies have changed Montana and the world, and express how those technologies influence, and are influenced by, cultural practices |
| **CS.IC.5.2** | identify ways to improve the accessibility and usability of technology products for the diverse needs and wants of users |

| **CS.5** | **COMPUTER SCIENCE CONTENT STANDARDS FOR FIFTH GRADE** |
|---|---|
| **CS.IC.5.3** | utilize diverse perspectives for the purpose of improving computational artifacts |
| **CS.IC.5.4** | apply laws associated with digital information and intellectual property |
| **CS.IC.5.5** | describe ethical issues that relate to computing devices and networks |
| **CS.NI.5** | **Computer science networks and the internet standards for fifth grade are that each student will** |
| **CS.NI.5.1** | explain cybersecurity problems |
| **CS.NI.5.2** | explain how personal information can be protected |

## SIXTH-EIGHTH GRADES

### STANDARDS

| CS.6-8 | COMPUTER SCIENCE CONTENT STANDARDS FOR SIXTH THROUGH EIGHTH GRADE |
|---|---|
| **CS.AP.6-8** | **Computer science algorithms and programming standards for sixth through eighth grades are that each student will** |
| **CS.AP.6-8.1** | use algorithms to address complex problems |
| **CS.AP.6-8.2** | create clearly named variables that represent different data types and perform operations on their values |
| **CS.AP.6-8.3** | develop programs that combine control structures, including nested loops and compound conditionals |
| **CS.AP.6-8.4** | decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs |
| **CS.AP.6-8.5** | create procedures with parameters to organize code and make it easier to reuse |
| **CS.AP.6-8.6** | seek and incorporate feedback from team members and users to refine a solution that meets user needs |
| **CS.AP.6-8.7** | incorporate existing code, media, and libraries into original programs, and give attribution |
| **CS.AP.6-8.8** | systematically test and refine programs using a range of test cases |
| **CS.AP.6-8.9** | distribute tasks and maintain a project timeline when collaboratively developing computational artifacts |
| **CS.AP.6-8.10** | document programs in order to make them easier to follow, test, and debug |
| **CS.CS.6-8** | **Computer science computing systems standards for sixth through eighth grades are that each student will** |
| **CS.CS.6-8.1** | recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices |
| **CS.CS.6-8.2** | design projects that combine hardware and software components to collect and exchange data |
| **CS.CS.6-8.3** | systematically identify and fix problems with computing devices and their components |
| **CS.DA.6-8** | **Computer science data and analysis standards for sixth through eighth grades are that each student will** |

| CS.6-8 | COMPUTER SCIENCE CONTENT STANDARDS FOR SIXTH THROUGH EIGHTH GRADE |
|---|---|
| **CS.DA.6-8.1** | collect data using computational tools and transform the data to make it more useful and reliable |
| **CS.DA.6-8.2** | represent data using multiple formats |
| **CS.DA.6-8.3** | refine computational models based on the data they have generated |
| **CS.IC.6-8** | **Computer science impacts of computing standards for sixth through eighth grades are that each student will** |
| **CS.IC.6-8.1** | compare tradeoffs associated with computing technologies that affect people's everyday activities and career options in Montana and the world, urban, rural, and reservation communities |
| **CS.IC.6-8.2** | discuss issues of bias and accessibility in the design of existing technologies |
| **CS.IC.6-8.3** | collaborate with other contributors when creating a computational artifact |
| **CS.IC.6-8.4** | describe tradeoffs between allowing information, personal or intellectual, to be public and keeping information private and secure |
| **CS.NI.6-8** | **Computer science networks and the internet standards for sixth through eighth grades are that each student will** |
| **CS.NI.6-8.1** | explain how physical and digital security measures protect electronic information |
| **CS.NI.6-8.2** | apply multiple methods of encryption to demonstrate how to securely transmit information |
| **CS.NI.6-8.3** | demonstrate how information is broken down and transmitted through multiple devices over networks and the internet and reassembled at the destination |

## NINTH-TWELFTH GRADES

### STANDARDS

| CS.9-12 | COMPUTER SCIENCE CONTENT STANDARDS FOR NINTH THROUGH TWELFTH GRADE |
|---|---|
| **CS.AP.9-12** | **Computer science algorithms and programming standards for ninth through twelfth grades are that each student will** |
| **CS.AP.9-12.1** | create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests |
| **CS.AP.9-12.2** | describe how artificial intelligence drives many software and physical systems |
| **CS.AP.9-12.3** | implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem |
| **CS.AP.9-12.4** | use and adapt classic algorithms to solve computational problems |
| **CS.AP.9-12.5** | evaluate algorithms in terms of their efficiency, correctness, and clarity |
| **CS.AP.9-12.6** | use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables |
| **CS.AP.9-12.7** | compare and contrast fundamental data structures and their uses |
| **CS.AP.9-12.8** | justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made |
| **CS.AP.9-12.9** | design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions |
| **CS.AP.9-12.10** | decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, or objects |
| **CS.AP.9-12.11** | create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs |

| CS.9-12 | COMPUTER SCIENCE CONTENT STANDARDS FOR NINTH THROUGH TWELFTH GRADE |
|---|---|
| **CS.AP.9-12.12** | construct solutions to problems using student-created procedures, modules, or objects |
| **CS.AP.9-12.13** | analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution |
| **CS.AP.9-12.14** | demonstrate code reuse by creating programming solutions using libraries and application programming interfaces |
| **CS.AP.9-12.15** | systematically design and develop programs for broad audiences by incorporating feedback from users |
| **CS.AP.9-12.16** | evaluate and refine computational artifacts to make them more usable and accessible |
| **CS.AP.9-12.17** | design and develop computational artifacts working in team roles using collaborative tools |
| **CS.AP.9-12.18** | document design decisions using text, graphics, presentations, or demonstrations in the development of complex programs |
| **CS.AP.9-12.19** | plan and develop programs for broad audiences using a software life cycle process |
| **CS.AP.9-12.20** | explain security issues that might lead to compromised computer programs |
| **CS.AP.9-12.21** | develop programs for multiple computing platforms |
| **CS.AP.9-12.22** | use version control systems, integrated development environments, and collaborative tools and practices in a group software project |
| **CS.AP.9-12.23** | develop and use a series of test cases to verify that a program performs according to its design specifications |
| **CS.AP.9-12.24** | modify an existing program to add additional functionality and discuss intended and unintended implications |
| **CS.AP.9-12.25** | evaluate key qualities of a program through a process such as a code review |

| CS.9-12 | COMPUTER SCIENCE CONTENT STANDARDS FOR NINTH THROUGH TWELFTH GRADE |
|---|---|
| **CS.AP.9-12.26** | compare multiple programming languages and discuss how their features make them suitable for solving different types of problems |
| **CS.CS.9-12** | **Computer science computing systems standards for ninth through twelfth grades are that each student will** |
| **CS.CS.9-12.1** | explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects |
| **CS.CS.9-12.2** | compare levels of abstraction and interactions between application software, system software, and hardware layers |
| **CS.CS.9-12.3** | categorize the roles of operating system software |
| **CS.CS.9-12.4** | develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors |
| **CS.CS.9-12.5** | illustrate ways computing systems implement logic, input, and output through hardware components |
| **CS.DA.9-12** | **Computer science data and analysis standards for ninth through twelfth grades are that each student will** |
| **CS.DA.9-12.1** | create interactive data visualizations using software tools to help others better understand authentic phenomena |
| **CS.DA.9-12.2** | use data analysis tools and techniques to identify patterns in data representing complex systems |
| **CS.DA.9-12.3** | select data collection tools and techniques to generate data sets that support a claim or communicate information |
| **CS.DA.9-12.4** | translate between different bit representations of authentic phenomena, including characters, numbers, and images |
| **CS.DA.9-12.5** | evaluate the tradeoffs in how data elements are organized and where data is stored |
| **CS.DA.9-12.6** | create computational models that represent the relationships among different elements of data collected from a phenomenon or process |

| CS.9-12 | COMPUTER SCIENCE CONTENT STANDARDS FOR NINTH THROUGH TWELFTH GRADE |
|---|---|
| **CS.DA.9-12.7** | evaluate the ability of models and simulations to test and support the refinement of hypotheses |
| **CS.IC.9-12** | **Computer science impacts of computing standards for ninth through twelfth grades are that each student will** |
| **CS.IC.9-12.1** | evaluate the ways computing technologies, globally and locally impact personal, ethical, social, economic, and cultural practices |
| **CS.IC.9-12.2** | evaluate the ways computing technologies impact American Indian communities in Montana |
| **CS.IC.9-12.3** | test and refine computational artifacts to reduce bias and equity deficits |
| **CS.IC.9-12.4** | demonstrate ways a given algorithm applies to problems across disciplines |
| **CS.IC.9-12.5** | evaluate computational artifacts to maximize their beneficial effects and minimize harmful effects on society |
| **CS.IC.9-12.6** | evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society, including the impact on American Indians living in urban, rural, and reservation communities |
| **CS.IC.9-12.7** | predict how computational innovations that have revolutionized aspects of our culture might evolve |
| **CS.IC.9-12.8** | use tools and methods to connect and work with others on a project including people in different cultures and career fields |
| **CS.IC.9-12.9** | explain the beneficial and harmful effects that intellectual property laws can have on innovation |
| **CS.IC.9-12.10** | explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users |
| **CS.IC.9-12.11** | evaluate the social and economic implications of privacy in the context of safety, law, or ethics |

| CS.9-12 | COMPUTER SCIENCE CONTENT STANDARDS FOR NINTH THROUGH TWELFTH GRADE |
|---|---|
| **CS.IC.9-12.12** | debate laws and regulations that impact the development and use of software |
| **CS.NI.9-12** | **Computer science networks and the internet standards for ninth through twelfth grades are that each student will** |
| **CS.NI.9-12.1** | recommend security measures to address various scenarios based on factors including efficiency, feasibility, and ethical impacts |
| **CS.NI.9-12.2** | explain tradeoffs when selecting and implementing cybersecurity recommendations |
| **CS.NI.9-12.3** | compare ways software developers protect devices and information from unauthorized access |
| **CS.NI.9-12.4** | evaluate the scalability and reliability of networks by describing the relationship between routers, addressing, switches, servers, and topology |
| **CS.NI.9-12.5** | give examples to illustrate how sensitive data can be affected by malware and other attacks |
| **CS.NI.9-12.6** | compare various security measures, considering tradeoffs between the usability and security of a computing system |
| **CS.NI.9-12.7** | discuss the issues that impact functionality |

## ADAPTING CURRICULUM TO MEET THE NEEDS OF EXCEPTIONAL LEARNERS

Good curriculum must respect the unique characteristics of the learner. It should recognize and support the need of each learner to make sense of ideas and information, reconstructing older understandings with new ones. Good curriculum will address interest and readiness levels. When implementing the Montana Content Standards, schools must provide *all* students with appropriate challenges. In the words of Carol Ann Tomlinson in the foreword to her book *How to Differentiate Instruction in a Mixed-Ability Classroom,*

> Acknowledging that students learn at different speeds and that they differ widely in their ability to think abstractly or understand complex ideas is like acknowledging that students at any given age aren't the same height: It is not a statement of worth, but of reality. To accommodate this reality, teachers can create "user-friendly" environment, one in which they flexibly adapt pacing, approaches to learning and channels for expressing learning in response to their students' differing needs. While the goal of each student is challenge and substantial growth, teachers must often define challenge and growth differently in response to students' varying interests and readiness levels (Tomlinson, 2001).

All learning happens on a continuum from novice to expert. Each stage of the continuum has different learning characteristics and learning needs. In The Parallel Curriculum Model, this continuum is called Ascending Levels of Intellectual Demand (ALID), and the learner characteristics are described as:

Novice

- Experiences content at a concrete level
- Manipulates micro-concepts one-at-a-time
- Needs skills instruction and guided practice
- Requires support, encouragement and guidance
- Seeks affirmation of competency in order to complete a task

Apprentice

- Understands the connections among micro-concepts within the discipline
- Connects information within a micro-concept
- Begins to interpret generalizations and themes that connect concepts
- Applies skills with limited supervision

- Seeks confirmation at the end of the task
- Reflects upon content and skills when prompted

Practitioner

- Manipulates two or more micro-concepts simultaneously
- Creates generalizations that explain connections among concepts
- Selects and utilizes skills in order to complete a task
- Seeks input from others as needed
- Exhibits task commitment and persistence when challenges are moderate
- Reflects upon both content and skills in order to improve understanding and performance

Expert

- Utilizes concepts with and among disciplines in order to derive theories and principles
- Creates innovations within a field
- Practices skill development independently and for the purpose of improvement
- Seeks input from other experts in a field for a specific purpose
- Works to achieve flow and derives pleasure from the experience (high challenge, advanced skill/ knowledge)
- Independent and self-directed
- Seeks experiences that cause a return to previous levels in varying degrees

This information guides curriculum design and instructional delivery by articulating the changes that characterize each learner at the stages between novice and expert. It provides a framework for thinking about how to challenge each learner with incremental sophistication—where each learning experience is just above easy reach of the learner who remains challenged and engaged. With a clear understanding of the characteristics and needs of each learner, the teacher can select assessment tools, interpret assessment data with accuracy and use the data to create responsive curriculum and instruction. Teachers can use scaffolding techniques and instructional strategies appropriate to the needs of each student (Tomlinson, Parallel curriculum model: a design to develop high potential and challenge high-ability learners).

## MODEL CURRICULUM MACRO- OR OVERARCHING CONCEPTS

Change

- Change generates additional change
- Change can be either positive or negative
- Change is inevitable
- Change is necessary for growth
- Change can be evolutionary or revolutionary

Conflict

- Conflict is composed of opposing forces or needs
- Conflict may be natural or human made
- Conflict may be intentional or unintentional
- Conflict may allow for synthesis and change
- Conflict can inspire fights or flight
- Conflict may inspire interest and attention

Exploration

- Exploration requires recognizing purpose and responding to it
- Exploration confronts “the unknown”
- Exploration may result in “new findings” or the confirmation of “old findings”

Force or Influence

- Force attracts, holds or repels
- Force affects or changes
- Force and inertia are co-dependent
- Force may be countered with equal or greater force

Order or Chaos

- Order may be natural or constructed
- Order may allow for prediction
- Order is a form of communication
- Order may have repeated patterns
- Order and chaos are reciprocals
- Order leads to chaos and chaos leads to order

Patterns

- Patterns have segments that are repeated
- Patterns allow for prediction
- Patterns have internal order
- Patterns are enablers
- Patterns have limits

Power

- Power is the ability to influence
- Power may be used or abused

- Power is always present in some form
- Power may take many forms (chemical, electrical, mechanical, spiritual, political)

Structure

- Structures have parts that interrelate
- Parts of structures support and are supported by other parts
- Smaller structures may be combined to form larger structures
- A structure is no stronger than its weakest component parts

Systems

- Systems have parts that work to complete a task
- Systems are composed of sub-systems
- Parts of systems are interdependent upon one another and form symbiotic relationships
- A system may be influenced by another system
- System interaction
- Systems follow rules

Relationships

- Everything is related in some way
- All relationships are purposeful
- Relationships change over time

(Curry)

# INSTRUCTIONAL MATERIALS SELECTION GUIDANCE

## SELECTION GUIDES AND CRITERIA

- Guide for Selecting Materials Aligned to Montana's Content Standards (word version)
- Criteria for Selecting Materials (word version) guides.
- Evaluating American Indian Materials and Resources for the Classroom

## QUESTIONS TRUSTEES SHOULD ASK

**Before the adoption process**

- What are the state guidelines and recommendations for curricular and instructional matters?
- How long has it been since the last time new materials were selected in each subject area?
- What is the budget for new materials?
- What is the timeline for selection?
- What is the district's instructional vision?
- What community values, needs, and curricular concerns should we be aware of?
- How will you engage stakeholders?
- Who will be on the adoption committee?
- How can you ensure diverse perspectives in the selection process?
- How will the adoption process be communicated to all stakeholders?
- How will you determine alignment to college and career-ready standards?
- Will you be using external or independent vetting such as EdReports.org?
- What is the plan for curriculum rollout, implementation, and professional learning?

("How School Boards . . .")

# GLOSSARY

Visit the Computer Science Teachers' Association glossary (csteachers.org/page/glossary) to access a comprehensive glossary of terms used in the Montana Content Standards for Computer Science.

# WORKS CITED

"Among colleagues: what do teachers need from curriculum guides?" Educational Leadership September 2010: 96.

Bransford, John D. How people learn: brain, mind, experience and school. Washington, D.C.: The National Academies Press, 1999.

Curry, J. Curriculum Guide or the Education of Gifted High School Students. Retrieved 12,1, 2010, from www.eric.ed.gov: 1993.

Donovan, M. Suzanne. How people learn: history, mathematics and science in the classroom. Washington, D.C.: The National Academies Press, 2005.

Erickson, H. Lynn. Concept-based curriculum and instruction for the thinking classroom. Thousand Oaks, CA: Corwin Press, 2007.

"How School Boards Can Support Districts to Adopt Quality Instructional Materials." EdReports, edreports.org/resources/article/how-school-boards-can-support-districts-to-adopt-quality-instructional-materials.

Lezotte, Larry "What Do Effective Schools Do?" Effective Schools, www.effectiveschools.com/#.

Tomlinson, Carol Ann. The parallel curriculum: a design to develop learner potential and challenge advanced learners. Thousand Oaks, CA: Corwin Press, 2009.

Tomlinson, C. A. (2001). *How to differentiate instruction in a mixed-ability classroom.* Alexandria, VA: Association for Supervision and Curriculum Development.

Tomlinson, C. A. (2009). *The parallel curriculum: a design to develop learner potential and challenge advanced learners.* Thousand Oaks, CA: Corwin Press.

Wang, H.A., and William Schmidt. "History, philosophy and sociology of science in science education: results from the third international mathematics and science study." Science & Education (2001): 51-70.

Wiggins, Grant and Jay McTighe. Understanding by Design. Expanded 2nd Ed. Alexandria: ASCD, 2005.
—. Understanding by Design Professional Development Workbook. Alexandria, VA: ASCD, 2004.

# APPENDIX A: RESOURCES

**Alice**

Alice (alice.org) is a block-based programming environment that makes it easy to create animations, build interactive narratives, or program simple games in 3D. Alice is designed to teach logical and computational thinking skills, fundamental principles of programming, and to be a first exposure to object-oriented programming. The Alice website also includes a Resources section (alice.org/resources) with lesson plans, exercises, projects, and curriculum materials.

- Alice 2: Teach fundamental programming skills and computational thinking. Alice 2 can be used with grades 3-6 or as an introduction to block-based programming for grades 6-12.
- Alice 3: With an added emphasis on object-oriented concepts, Alice 3 has a new rich gallery of models that includes everything you need to spark your creativity including a full Sims ™ character builder. It can serve as a great early learning tool as well as extending to assist in a full transition to the Java programming language. Grades 6-12.

Adventures in Alice Programming (cs.duke.edu/csed/alice/aliceInSchools) from Duke University is a project for integrating the programming language Alice into middle schools and high schools. The site includes free materials, lesson plans (multiple subjects, grades 3-12), tutorials, links to past workshop materials and presentations, and a link to a Coursera course entitled Introduction to Programming and Animation with Alice that launched in January 2020.

**App Inventor**

Build apps with App Inventor from MIT (appinventor.mit.edu); the completed applications only work on Android devices, a separate iOS compatible version is also available on the Apple App Store (apps.apple.com/us/app/mit-app-inventor/id1422709355).

- Includes resources for teachers (appinventor.mit.edu/explore/teach) to teach Computational Thinking and Mobile Computer Science Principles.
- Multiple tutorials (appinventor.mit.edu/explore/ai2/tutorials) available for students and teachers to build apps.
- AI with MIT App Inventor (appinventor.mit.edu/explore/ai-with-mit-app-inventor) includes tutorial lessons as well as suggestions for student explorations and project work.

Youth Apps Challenge (technology-alliance.com/teacher-resources) uses both Scratch and App Inventor.

- Also includes link to App Inventor 101 course (canvas.instructure.com/courses/957451)

- Youth Apps Challenge Curriculum guide, Version 5 (static1.squarespace.com/static/545b1745e4b0a4696b7278fd/t/563ba707e4b0d1ab5805a3e5/1446749959693/Youth+Apps+Challenge+v5_interactive.pdf)

**Apple**
Students learn to code with Swift and Xcode12. Swift works on iOS devices, including iPhones, iPads, and Mac computers and MacBooks. It includes free curriculum materials for students and teachers.(apple.com/education/k12/teaching-code)

Develop in Swift Professional Learning is a free online professional learning course available on the Canvas Learning Management System to prepare educators to teach Develop in Swift Explorations with optional content for Develop in Swift AP® CS Principles. (appleeducation.catalog.instructure.com/courses/53455/enrollment/new)

**Bootstrap**
Bootstrap (bootstrapworld.org) provides researched-based curriculum for grades 6-12. The materials reinforce core concepts in math, enabling non-CS teachers to provide rigorous and engaging computing content. Bootstrap offers resource for teaching computer science concepts and practices in math and science subjects.

**CMU Computer Science Academy**
CMU CS Academy (academy.cs.cmu.edu) is an online, graphics-based computer science curriculum taught in Python provided by Carnegie Mellon University and it is entirely free. CMU CS Academy offers three courses:

- CS1: The Flagship course, the curriculum is a deep dive into the fundamentals of programming concepts and teaches text-based coding using Python. Grades 8-12.
- CS0: A lighter version of our CS1 curriculum, it is designed to engage and excite future CS1 students. This curriculum is taught using text-based coding in Python. Grades 5-8.
- AP CSP: The course is an extension of Code.org's AP Computer Science Principles (CSP) prep course. The course utilizes Code.org’s AP CSP materials for all of the units except for the programming units, which use javascript. For the programming units, students and teachers will work from the CMU CS Academy platform and program in Python.

**CodeAcademy**

CodeAcademy: codecademy.com/

- Resources for Web Development (HTML/CSS; JavaScript), Programming and Computer Science (Python, CMD Online), Data Science (Python, SQL), and many more.

**CodeHS**

Courses for both middle school and high school: Codehs.com

- Options include Block Programming, Cybersecurity, Python, Java, AP Computer Science Principals, Web Design, and Interdisciplinary Courses for Math and Science.
- Offers a complete interdisciplinary collection of courses to integrate computer science into existing courses at both the middle school and high school levels.

**Code.org**

Courses on code.org for students grades K-12 and professional learning for teachers.

- Pre-reader Express: Grades K-2
- Computer Science Fundamentals: Grades K-5
- Computer Science Fundamentals Express: Ages 8-18
- Computer Science Discoveries: Grades 6-10
- Computer Science Principals: Grades 9-12

**Computer Science Teachers Association**

The Computer Science Teachers Association (CSTA) (cstechers.org) is a professional organization for K-12 computer science teachers. CSTA provides resources, standards materials, and professional development opportunities. Teachers can join CTSA for free with a CSTA Basic Membership.

- Join CTSA: csteachers.org/page/individual-membership
- Standards Resources: https://www.csteachers.org/page/about-csta-s-k-12-nbsp-standards
- Professional Development Opportunities/Online Teacher Learning: csteachers.org/page/quality-pd

**CS Unplugged**

CS Unplugged (csunplugged.org/en) is a collection of free teaching materials to teach Computer Science through engaging games and puzzles using cards, string, crayons, and lots of running around. Activities are designed for students K-5; however, they can also be used as introductory activities for grades 6-12. PDF of the 2015 edition of the *CS Unplugged* book: classic.csunplugged.org/books

**CYBER.org**

CYBER.ORG (cyber.org) empowers educators to teach cyber confidently, resulting in students with the skills and passion needed to succeed in the cyber workforce. Courses are available for grades 2-12 and CYBER.org's find curricula feature allows teachers to search for materials by type (unit, course, or activity), grade level, and/or subject. CYBER.org offers free, web-based curriculum in a number of subject areas including

- Career & Technical Education
- Coding
- Computer Science
- Cybersecurity
- Humanities
- Mathematics
- Robotics
- Science

**Google for Education**

CS First: csfirst.withgoogle.com/s/en/home

- Lessons and resources for ages 9-14

Code with Google: edu.google.com/code-with-google/

- Includes CS First and learning JavaScript with Grasshopper to develop apps.

**Hour of Code**

Each year, Hour of Code (hourofcode.com/us/learn) is held internationally during Computer Science Week in December. Many organizations participate and have resources available year-round on their websites. For both students and teachers new to computer science and/or coding, the projects are great for learning and exploring.

**LEGO**

Coding for Kids: lego.com/en-us/categories/coding-for-kids
Gaming: lego.com/en-us/categories/gaming
LEGO Education BricQ Motion: education.lego.com/en-us/?icmp=LP-SHSB-Standard-NO_Sidekick_LEGO_Education_Website_LP-TH-NO-IW4DYKOV0T

**Microsoft EDU**

Training resources for both students and teachers: education.microsoft.com/en-us

- Minecraft, Education Edition: education.minecraft.net/
- Minecraft, Education Edition Coding Academy: education.microsoft.com/en-us/learningPath/d1f49722
- Minecraft, Education Edition Teacher Academy: education.microsoft.com/en-us/learningPath/3eede2ae
- Microsoft MakeCode: microsoft.com/en-us/makecode
- MakeCode for Mindcraft Video Tutorials Grades 6-8: education.microsoft.com/en-us/lesson/d7c5544a
- MakeCode for Micro.bit Video Tutorials Grades 6-8: education.microsoft.com/en-us/lesson/9fcefa98
- MakeCode Arcade Video Tutorials Grades 6-12: education.microsoft.com/en-us/lesson/06de967f
- Learn Xamarian: dotnet.microsoft.com/learn/xamarin
    - Free courses, tutorials, videos, and more for learning mobile development with Xamarin.

**Scratch Resources**

- Scratch for Educators (scratch.mit.edu/educators/): Available resources include Educator Guides, Scratch Tutorials, Coding Cards, and Ideas Pages
- Creative Computing Curriculum: A collection of ideas, strategies, and activities for an introductory creative computing experience using Scratch designed by the Creative Computing Lab at the Harvard Graduate School of Education. Creative Computing Curriculum materials can be downloaded as Google Slides/PowerPoint to allow for editing and classroom personalization. (scratched.gse.harvard.edu/guide/index.html)
- Scratch Projects: A Comprehensive Course Lesson Plan for Version 3 (scratched.gse.harvard.edu/sites/default/files/lp_scratch3.pdf)
- Computational Thinking with Scratch (scratched.gse.harvard.edu/ct/index.html)
- Introduction to Programming with Scratch (designed for educators)( csed.uni.edu/scratch-intro/)
- ScratchED (scratched.gse.harvard.edu/)
- Scratch Introductory Course from Redware (scratch.redware.com/scratch/overview)
- ScratchJr: Designed for early elementary grades, K-2. (scratchjr.org)

**Unity**

- Unity (unity.com) can be used for teaching & learning coding fundamentals for 2D, 3D, VR & game development.
- Unity Learn (learn.unity.com/educators) platform provides access to over 750 hours of on-demand and live learning resources to help bring visions to life.
- Unity Essentials (learn.unity.com/pathway/unity-essentials): Designed for anyone new to Unity grades 7-12.
- Unity Junior Programmer (learn.unity.com/pathway/junior-programmer): Designed for those wanting to learn to code in Unity grades 9-12. Includes 3D animation, scripting, XR development, game development, and video animation.

0 copies of this public document were published at an estimated cost of $0 per copy, for a total cost of $0, which includes $0 for printing and $0 for distribution.

The OPI is committed to providing reasonable accommodations to people with disabilities. If you need a reasonable accommodation, require an alternate format, or have questions concerning accessibility, contact the OPI ADA Coordinator, 406-444-3161, opiada@mt.gov, or call 711 for assistance.