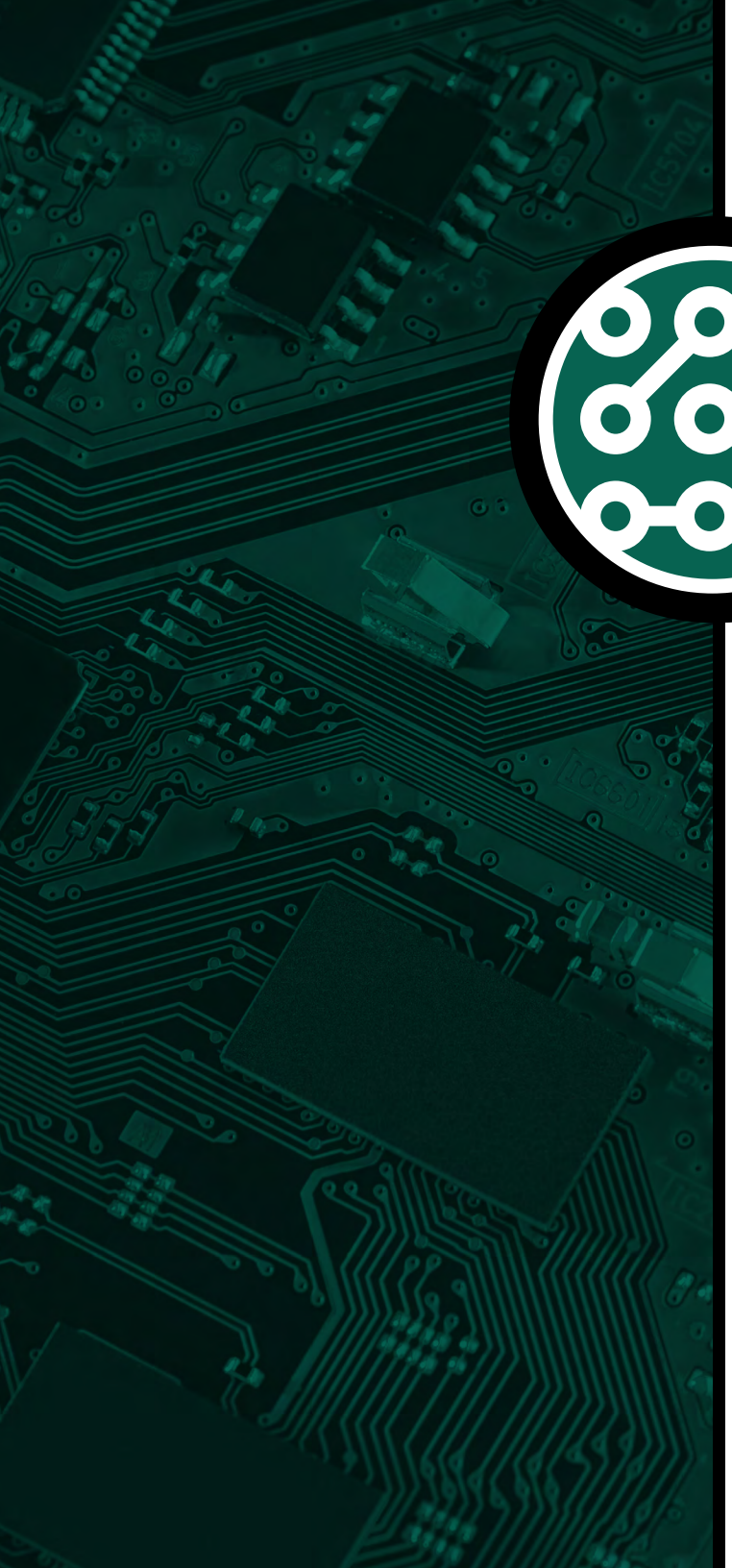

Oklahoma Academic Standards

COMPUTER SCIENCE

DRAFT

OKLAHOMA
Education

Table of Contents

Introduction

The Oklahoma Academic Standards for Computer Science (OAS-CS) specify what students should know and be able to do as learners in their discipline at the end of each grade level or course. Students have different levels of experience within a discipline so teachers can attend to both grade-level standards and meet the individual needs of students who may be performing at levels above or below grade level. The order of the standards at any grade level is not meant to imply a sequence of topics and should be considered flexible for the organization of any course. The document provides specific grade-level standards for students in grades kindergarten through grade 12.

Computer science core concepts and practices in the OAS-CS are aligned vertically and are grade-level specific to ensure proper scaffolding of content knowledge and skills. The structure of the OAS-CS allows for a variety of instructional methods, such as through integration or computer science-specific courses. Grades K-8 and Level 1 standards are for all Oklahoma students, while the Level 2 standards are designed for those students intending to specialize in computer science careers or higher education pathways.

The Oklahoma Academic Standards for Computer Science Review was informed by the 2018 Oklahoma Academic Standards for Computer Science, other states' standards documents and curriculum framework guides, and some of the latest research in computer science education. The Oklahoma Academic Standards for Computer Science are neither curriculum nor instructional practices; standards serve as a foundation for curriculum.

- Standards: The concepts, content, and skills in which students should gain proficiency.
- Curriculum: The materials and resources used for teaching the standards.
- Instruction: The practices teachers use to deliver academic content to students. Teachers should utilize a variety of instructional techniques and strategies to ensure students master academic standards.

Defining Computer Science

Computer science is the study of computers and algorithmic processes and is often confused with general computer use and computer applications.

Computer Science Education Includes:	Computer Science Education Does Not Include:
• Study of computers • Analyzing and creating programming/software designs • Studying hardware designs • Researching and analyzing usage in local and national context • Evaluating the impact on society	• Teaching students to type or use a mouse • Learning to use applications (e.g., word processing program or slides) • Playing video games • Building or repairing a computer

Computer Science Practices

The Oklahoma Computer Science Practices capture the computer science experience of Oklahoma students as they pursue computer science literacy. These computer science practices describe the processes and approaches that computationally literate students use to fully engage in a data-rich and interconnected world. The practices are not delineated by grade bands. Rather, the practices describe how students should develop each practice throughout their educational experiences. Suggested computer science practices are connected to the Oklahoma Academic Standards for Computer Science to support students in developing a deep understanding of the standards and concepts.

Foster an Inclusive Computing Culture

Understand the unique contexts in which people operate and consider the needs of diverse users during the design process. Students will include the unique perspectives of others and reflect on one's own perspectives when designing and developing computational products, address the needs of diverse end users to produce artifacts with broad accessibility and usability, and employ self- and peer-advocacy to meet the needs of all potential end users (including themselves).

Collaborate Around Computing

Perform a computational task by working in pairs and on teams. Students will cultivate working relationships with individuals possessing diverse perspectives, skills, and personalities, create team norms and expectations, solicit and incorporate feedback, and evaluate and select technological tools that can be used to collaborate on a computer science project.

Recognizing and Defining Computational Problems

Recognize appropriate and worthwhile opportunities to apply computation. Students will work to solve a problem by defining the problem, breaking it down into parts, and evaluating each part to determine whether a computational solution is appropriate.

Developing and Using Abstractions

Identify patterns and extract common features from specific examples to create generalizations. Students will manage complexity by using generalized solutions and parts of solutions designed for broad reuse to simplify the development process.

Creating Computational Artifacts

Develop computational artifacts 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

Use a process to test and refine a computational artifact. Students will engage in debugging (identifying and fixing errors) and comparing actual outcomes to intended outcomes. Students will also respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts.

Communicating about Computing

Express and exchange ideas with others. Students will 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 in multiple ways.

Computer Science Concepts and Subconcepts

A concept represents a specific area of importance in computer science. Each concept includes multiple subconcepts that represent the specific ideas within that concept. For example, the Computing Systems concept has three subconcepts: Devices, Hardware & Software, and Troubleshooting. There are five core concepts in the Oklahoma Academic Standards for Computer Science:

1. **Computing Systems:** Students should understand the hardware and software components of computing systems and devices. This concept includes exploring how hardware and software components work together to be functional for a user and discovering troubleshooting techniques to solve software and hardware problems.
2. **Networks and the Internet:** Students should understand how computers communicate with other computers and exchange data between devices. Networks and communication systems provide greater connectivity in the computing world by providing fast, secure communication. This concept includes exploring cybersecurity principles to ensure safe online experiences.
3. **Data Analysis:** Students should understand how to leverage computing systems to formulate questions that can be addressed with data, and to collect, organize, and display relevant data to answer them. This concept includes selecting and using different types of data and using this data to make inferences and predictions.
4. **Algorithms and Programming:** Students should understand algorithms as sequences of steps designed to accomplish specific tasks. Students will explore the role of algorithms in providing instructions for computing devices. This concept includes breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.
5. **Impacts of Computing:** Students should understand how individual computing choices impact others positively and negatively at local, national, and global levels. This concept includes exploring safety, law, and ethics related to using technology and how technology impacts cultures and future generations.

Computing Systems						
Subconcept	Kindergarten (K)	First Grade (1)	Second Grade (2)	Third Grade (3)	Fourth Grade (4)	Fifth Grade (5)
Devices	K.CS.D.01 With guidance, follow directions and start to make appropriate choices to use computing devices to perform a variety of tasks.	1.CS.D.01 With guidance, select and use a computing device to perform a variety of tasks for an intended outcome.	2.CS.D.01 Select and use appropriate computing devices or software to perform a variety of tasks for an intended outcome.	3.CS.D.01 Select and use computing systems to perform a variety of tasks for an intended outcome.	4.CS.D.01 Select and use appropriate computing systems to perform a variety of tasks for an intended outcome while recognizing that users have different needs for the technology they use.	5.CS.D.01 Select and use the most efficient computing systems to perform a variety of tasks for an intended outcome while recognizing that users have different needs for the technology they use.
	Practice: Fostering an Inclusive Computing Culture					
Hardware & Software	K.CS.HS.01 Use appropriate terminology to locate and identify common computing devices and components in a variety of environments (e.g., desktop computer, laptop computer, tablet device, monitor, keyboard, mouse, printer).	1.CS.HS.01 Use appropriate terminology in naming and describing the function of common computing devices and components (e.g., mouse is used to control the cursor).	2.CS.HS.01 Identify the components of a computing system and what the basic functions are (e.g., hard drive and memory) as well as peripherals (e.g., printers, scanners, external hard drives) and external storage features and their uses (e.g., cloud storage).	3.CS.HS.01 Model how information flows through hardware and software to accomplish tasks.	4.CS.HS.01 Model that information is translated, transmitted, and processed in order to flow through hardware and software.	5.CS.HS.01 Model that information is translated into bits in order to transmit and process between hardware and software to accomplish tasks.
	Practice: Communicating about Computing			*Practice: Developing and Using Abstractions*		
Troubleshooting	K.CS.T.01 Recognize that computing systems might not work as expected and, with guidance, use accurate terminology to identify simple hardware or software problems (e.g., volume turned down on headphones, monitor turned off).	1.CS.T.01 Identify, using accurate terminology, simple hardware and software problems that may occur during use (e.g., program is not working as expected, no sound is coming from the device, caps lock turned on).	2.CS.T.01 Identify, using accurate terminology, simple hardware and software problems that may occur during use (e.g., program is not working as expected, no sound is coming from the device, caps lock turned on) and discuss	3.CS.T.01 Identify, using accurate terminology, simple hardware and software problems that may occur during everyday use, discuss problems with peers and adults, and apply strategies for solving these problems (e.g., refresh the screen, closing and	4.CS.T.01 Identify, using accurate terminology, simple hardware and software problems that may occur during everyday use, discuss problems with peers and adults, and apply strategies for solving these problems (e.g., rebooting the device, checking the	5.CS.T.01 Identify, using accurate terminology, simple hardware and software problems that may occur during everyday use. Discuss problems with peers and adults, apply strategies for solving these problems and explain why the strategies should work.

			problems with peers and adults.	reopening an application or file, unmuting or adjusting the volume on headphones).	power, force shut down of an application).	
	Practices: Testing and Refining Computational Artifacts, Communicating about Computing			*Practice: Testing and Refining Computational Artifacts*		
Networks & the Internet						
Subconcept	**Kindergarten**	**1st Grade**	**2nd Grade**	**3rd Grade**	**4th Grade**	**5th Grade**
Network Communication & Organization	K.NI.NCO.01 Recognize that computing devices can be connected together.	1.NI.NCO.01 Recognize that by connecting computing devices together, they can share information (e.g., remote storage, printing, the Internet).	2.NI.NCO.01 Recognize that computing devices can be connected in a variety of ways.	3.NI.NCO.01 Recognize that information is sent and received over physical or wireless paths.	4.NI.NCO.01 Explain how information is sent and received across physical or wireless paths (e.g., It is broken down into smaller pieces called packets and transmitted from one location to another).	5.NI.NCO.01 Model how information is broken down into packets (i.e., smaller pieces), transmitted through multiple devices over networks and the Internet, and reassembled at the destination.
	Practice: Developing and Using Abstractions					
Cybersecurity	K.NI.C.01 Discuss what passwords are and why we do not share them with others. With guidance, use passwords to access computing devices.	1.NI.C.01 Identify what passwords are; explain why they are not shared and discuss what makes a password strong. Independently, use passwords to access computing devices.	2.NI.C.01 Explain what passwords are, why we use them, and use strong passwords to protect computing devices and information from unauthorized access.	3.NI.C.01 Identify problems that relate to inappropriate use of computing devices and networks.	4.NI.C.01 Identify and explain issues related to responsible use of technology and information, and describe personal consequences of inappropriate use.	5.NI.C.01 Discuss real-world cybersecurity problems and identify strategies for how personal information can be protected.
	Practice: Communicating About Computing			*Practice: Recognizing and Defining Computational Problems*		
Data Analysis						
Subconcept	**Kindergarten**	**1st Grade**	**2nd Grade**	**3rd Grade**	**4th Grade**	**5th Grade**
Storage	K.DA.S.01 With guidance, locate, open, modify, and save an existing file with a computing device.	1.DA.S.01 With guidance locate, open, modify and save an existing file, use appropriate file-naming conventions, and recognize that the file exists within an	2.DA.S.01 With guidance, develop and modify an organizational structure by creating, copying, moving, and deleting files and folders.	3.DA.S.01 Recognize that different types of information are stored in different formats that have associated programs (e.g., documents open in a word processor) and	4.DA.S.01 Choose different storage locations (physical, shared, or cloud) based on the type of file, storage requirements (file size, availability,	5.DA.S.01 Evaluate trade-offs, including availability and quality, based on the type of file, storage requirements (e.g., file size, availability,

		organizational structure (e.g., drive, folder, file).		varied storage requirements.	available memory), and sharing requirements.	available memory), and sharing requirements.
	Practice: Developing and Using Abstractions					
Collection, Visualization, & Transformation	K.DA.CVT.01 With guidance, collect data and present it visually.	1.DA.CVT.01 With guidance, collect data and present it two different ways.	2.DA.CVT.01 With guidance, collect and present the same data in various visual formats.	3.DA.CVT.01 Collect and organize data in various visual formats.	4.DA.CVT.01 Organize and present collected data visually to highlight comparisons.	5.DA.CVT.01 Organize and present collected data to highlight comparisons and support a claim.
	Practices: Communicating About Computing, Developing and Using Abstractions			*Practice: Communicating About Computing*		
Inference & Models	K.DA.IM.01 With guidance, draw conclusions based on pictographs, real-object graphs, or patterns.	1.DA.IM.01 With guidance, identify and interpret data from a chart, bar graph, or pictograph (visualization) in order to draw conclusions, with or without a computing device.	2.DA.IM.01 With guidance, construct and interpret data with up to four categories and present it in a chart, bar graph, or pictograph (visualization) in order to draw conclusions with or without a computing device.	3.DA.IM.01 Utilize data to make predictions and discuss whether there is adequate data to make reliable predictions.	4.DA.IM.01 Utilize data to create models, answer investigative questions, and make predictions.	5.DA.IM.01 Determine how the accuracy of conclusions is influenced by the amount of data collected.
	Practice: Developing and Using Abstractions			*Practice: Communicating About Computing*		
Algorithms & Programming						
Subconcept	**Kindergarten**	**1st Grade**	**2nd Grade**	**3rd Grade**	**4th Grade**	**5th Grade**
Algorithms	K.AP.A.01 With guidance, model daily processes and follow algorithms (sets of step-by-step instructions) to complete tasks verbally, kinesthetically, with robot devices, or a programing language.	1.AP.A.01 With guidance, model daily processes and follow algorithms (sets of step-by-step instructions) to complete tasks verbally, kinesthetically, with robot devices, or a programing language.	2.AP.A.01 With guidance, model daily processes by creating and following algorithms (sets of step-by step instructions) to complete tasks verbally, kinesthetically, with robot devices, or a programing language.	3.AP.A.01 Model and compare multiple algorithms for the same task.	4.AP.A.01 Model, compare, and refine multiple algorithms for the same task.	5.AP.A.01 Model, compare and refine multiple algorithms for the same task and determine which is the most efficient.
	Practice: Developing and Using Abstractions			*Practices: Testing and Refining Computational Artifacts, Recognizing and Defining Computational Problems*		

Variables	K.AP.V.01 With guidance, recognize that computers represent different types of data using numbers or other symbols.	1.AP.V.01 With guidance, model the way that a program accesses stored data using a variable name.	2.AP.V.01 Model the way a computer program stores, accesses, and manipulates data that is represented as a variable.	3.AP.V.01 Create programs that use variables to store and modify grade level appropriate data.	4.AP.V.01 Create programs that use variables to store and modify grade level appropriate data.	5.AP.V.01 Create programs that use variables to store and modify grade level appropriate data.
	Practice: Developing and Using Abstractions			*Practice: Creating Computational Artifacts*		
Control	K.AP.C.01 With guidance, independently or collaboratively create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing (i.e., emphasizing the beginning, middle, and end).	1.AP.C.01 With guidance, independently or collaboratively create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing and repetition.	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 and repetition.	3.AP.C.01 Create programs using a programming language that utilize sequencing, repetition, conditionals, and variables to solve a problem or express ideas both independently and collaboratively.	4.AP.C.01 Create programs using a programming language that utilize sequencing, repetition, conditionals and variables using math operations manipulate values to solve a problem or express ideas both independently and collaboratively.	5.AP.C.01 Create programs using a programming language that utilize sequencing, repetition, conditionals, event handlers and variables using math operations to manipulate values to solve a problem or express ideas both independently and collaboratively.
	Practice: Creating Computational Artifacts					
Modularity	*Standards for this sub-concept begin in Third Grade.*			3.AP.M.01 Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.	4.AP.M.01 Decompose (break down) large problems into smaller, manageable subproblems to facilitate the program development process. *Practice(s): Recognizing and Defining Computational Problems*	5.AP.M.01 Decompose (break down) large problems into smaller, manageable subproblems and then into a precise sequence of instructions. *Practice(s): Recognizing and Defining Computational Problems*
				Practice: Recognizing and Defining Computational Problems		
				3.AP.M.02 With grade appropriate complexity, modify,	4.AP.M.02 With grade appropriate complexity, modify,	5.AP.M.02 With grade appropriate complexity, modify,

				remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.	remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.	remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.
				Practice: Creating Computational Artifacts		
Program Development	K.AP.PD.01 With guidance, create a grade-level appropriate artifact to illustrate thoughts, ideas, or stories in a sequential manner (e.g., story map, storyboard, and sequential graphic organizer).	1.AP.PD.01 Independently or with guidance, create a grade-level appropriate artifact to illustrate thoughts, ideas, or stories in a sequential (step-by-step) manner (e.g., story map, storyboard, and sequential graphic organizer).	2.AP.PD.01 Independently or with guidance, create a grade-level appropriate artifact to illustrate thoughts, ideas, or stories in a sequential manner (e.g., story map, storyboard, and sequential graphic organizer).	3.AP.PD.01 Use an iterative process to plan the development of a program while solving simple problems.	4.AP.PD.01 Use an iterative process to plan the development of a program that includes user preferences while solving simple problems.	5.AP.PD.01 Use an iterative process to plan the development of a program that includes others' perspectives and user preferences while solving simple problems.
	Practices: Creating Computational Artifacts, Communicating About Computing			*Practice: Creating Computational Artifacts*	*Practices: Fostering an Inclusive Computing Culture, Creating Computational Artifacts*	
	K.AP.PD.02 Independently or with guidance give credit to ideas, creations and solutions of others while developing algorithms.	1.AP.PD.02 Independently or with guidance give credit to ideas, creations and solutions of others while writing and/or developing programs.	2.AP.PD.02 Give credit to ideas, creations and solutions of others while writing and developing programs.	3.AP.PD.02 Observe intellectual property rights and give appropriate credit when creating or remixing programs.	4.AP.PD.02 Observe intellectual property rights and give appropriate credit when creating or remixing programs.	5.AP.PD.02 Observe intellectual property rights and give appropriate credit when creating or remixing programs.
	Practices: Communicating About Computing			*Practice(s): Creating Computational Artifacts, Communicating About Computing*		
	K.AP.PD.03 With guidance, independently or collaboratively debug algorithms using a programming language and/or	1.AP.PD.03 With guidance, independently or collaboratively debug programs using a programming language and/or unplugged activity that	2.AP.PD.03 With guidance, independently and collaboratively debug programs using a programming language and/or unplugged activity that	3.AP.PD.03 Analyze and debug a program that includes sequencing, repetition and variables in a programming language.	4.AP.PD.03 Analyze, create, and debug a program that includes sequencing, repetition, conditionals and variables in a programming language.	5.AP.PD.03 Analyze, create, and debug a program that includes sequencing, repetition, conditionals and variables in a programming language.

	unplugged activity that includes sequencing.	includes sequencing and repetition.	includes sequencing and repetition.			
	Practice(s): Testing and Refining Computational Artifacts					
	K.AP.PD.04 Use correct terminology (beginning, middle, end) in the development of an algorithm to solve a simple problem.	1.AP.PD.04 Use correct terminology (e.g., first, second, third) and explain the choices made in the development of an algorithm to solve a simple problem.	2.AP.PD.04 Use correct terminology (e.g., debug, program input/output, code) to explain the development of an algorithm to solve a problem in an unplugged activity, hands on manipulatives, or a programming language.	3.AP.PD.04 Communicate and explain program development choices using comments, presentations and demonstrations.	4.AP.PD.04 Communicate and explain program development choices using comments, presentations and demonstrations.	5.AP.PD.04 Communicate and explain program development choices using comments, presentations and demonstrations.
	Practice: Communicating About Computing					
Impacts of Computing						
Subconcept	**Kindergarten**	**1st Grade**	**2nd Grade**	**3rd Grade**	**4th Grade**	**5th Grade**
Culture	K.IC.C.01 Identify different ways in which types of technologies are used in your daily life.	1.IC.C.01 Identify how people use different types of technologies in their daily work and personal lives.	2.IC.C.01 Compare how people live and work before and after the implementation or adoption of new technology.	3.IC.C.01 Identify computing technologies that have changed the world, and express how those technologies influence and are influenced by cultural practices.	4.IC.C.01 Give examples of computing technologies that have changed the world, and express how those technologies influence and are influenced by cultural practices.	5.IC.C.01 Give examples and explain how computing technologies have changed the world, and express how computing technologies influence and are influenced by cultural practices within your community.
				Practice: Recognizing and Defining Computational Problems		
				3.IC.C.02 Consider users' backgrounds and ability levels and identify how computing devices have built-in features for increasing accessibility to these users.	4.IC.C.02 Consider a variety of users' backgrounds and needs to brainstorm ways to improve computing devices to increase accessibility to all users.	5.IC.C.02 Develop, test, and refine digital artifacts to improve accessibility and usability.

	Practice: Communicating About Computing			*Practice: Fostering an Inclusive Computing Culture*		
Social Interactions	K.IC.SI.01 With guidance, identify appropriate behavior while participating in an online environment.	1.IC.SI.01 With guidance, identify appropriate and inappropriate behavior, act responsibly, and know how to report concerns while participating in an online community.	2.IC.SI.01 With guidance, develop a code of conduct, explain, and practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior.	3.IC.SI.01 Develop a code of conduct, explain, and practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior.	4.IC.SI.01 Develop a code of conduct, explain, and practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior.	5.IC.SI.01 Develop a code of conduct, explain, and practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior.
	Practice: Collaborating Around Computing					
	Additional standards for this subconcept begin in Third Grade.			3.IC.SI.02 Identify how computational artifacts may be, or have been, improved to incorporate diverse perspectives.	4.IC.SI.02 As a team, consider each others' perspectives on improving a computational artifact.	5.IC.SI.02 As a team, collaborate with outside resources (other grade levels, online collaborative spaces) to include diverse perspectives to improve computational artifacts.
				Practice: Fostering an Inclusive Computing Culture		
Safety, Law, & Ethics	K.IC.SLE.01 With guidance, identify ways to stay safe online.	1.IC.SLE.01 Identify ways to stay safe online.	2.IC.SLE.01 Individually and collaboratively identify ways to stay safe online.	3.IC.SLE.01 Identify types of digital data that may have intellectual property rights that prevent copying or require attribution.	4.IC.SLE.01 Discuss the social impact of violating intellectual property rights.	5.IC.SLE.01 Observe intellectual property rights and give appropriate credit when using resources.
	Practice: Communicating About Computing					

Computing Systems					
Subconcept	**Sixth Grade (6)**	**Seventh Grade (7)**	**Eighth Grade (8)**	**Level 1 – By the end of Tenth Grade (L1)**	**Level 2 – By the end of Twelfth Grade (L2)**
Devices	6.CS.D.01 Evaluate existing computing devices and recommend improvements to the design based on personal interaction with the device.	7.CS.D.01 Evaluate existing computing devices and recommend improvements to the design based on how other users interact with the device.	8.CS.D.01 Develop and implement a process to evaluate existing computing devices and recommend improvements to the design based on how other users interact with the device.	L1.CS.D.01 Model how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.	*Students will continue to apply the standards from the previous grade levels.*
	Practice: Recognizing and Defining Computational Problems			*Practice: Developing and Using Abstractions*	
Hardware & Software	6.CS.HS.01 Model multiple methods of combining hardware and software to collect and exchange data.	7.CS.HS.01 Evaluate and recommend improvements to software and hardware combinations used to collect and exchange data.	8.CS.HS.01 Design and refine projects that combine hardware and software components to collect and exchange data.	L1.CS.HS.01 Analyze the levels of abstraction and interactions between application software, system software, and hardware.	L2.CS.HS.01 Identify and categorize the roles of a variety of operating system software.
	Practice: Creating Computational Artifacts			*Practice: Developing and Using Abstractions*	*Practice: Communicating About Computing*
Troubleshooting	6.CS.T.01 Identify and resolve software and hardware problems with computing devices and their components involving settings and connections.	7.CS.T.01 Identify and resolve complex software and hardware problems with computing devices and their components utilizing strategies such as developing and analyzing flow diagrams.	8.CS.T.01 Systematically identify, resolve, and document complex software and hardware problems with computing devices and their components.	L1.CS.T.01 Develop and apply criteria for the systematic discovery of errors and systematic strategies for the correction of errors in computing systems.	L2.CS.T.01 Illustrate how understanding the ways hardware components facilitate logic, input, output, and storage in computing systems will support troubleshooting.
	Practice: Testing and Refining Computational Artifacts				*Practice: Communicating About Computing*

Networks & the Internet					
Subconcept	**6th Grade**	**7th Grade**	**8th Grade**	**Level 1 – By the end of 10th Grade**	**Level 2 – By the end of 12th Grade**
Network Communication & Organization	6.NI.NCO.01 Model a simple protocol for transferring information using packets.	7.NI.NCO.01 Explain how a system responds when a packet is lost and the effect it	8.NI.NCO.01 Explain protocols and their importance to data transmission; model how packets are broken down into	L1.NI.NCO.01 Evaluate the scalability and reliability of networks by identifying and illustrating the basic components	L2.NI.NCO.01 Describe the issues that impact network functionality (e.g., bandwidth, load, latency, topology).

		has on the transferred information.	smaller pieces and how they are delivered.	of computer networks (e.g., routers, switches, servers, etc.) and network protocols (e.g., IP, DNS, etc.).	
	Practice): Developing and Using Abstractions				*Practice: Communicating About Computing*
Cybersecurity	6.NI.C.01 Identify existing cybersecurity concerns with the Internet and systems it uses.	7.NI.C.01 Explain how to protect electronic information, both physical (e.g., hard drive) and digital; identify cybersecurity concerns and options to address issues with the Internet and the systems it uses.	8.NI.C.01 Evaluate physical and digital procedures that could be implemented to protect electronic data/information; explain the impacts of cybersecurity threats and attacks.	L1.NI.C.01 Compare physical and cybersecurity measures by evaluating trade-offs between the usability and security of a computing system and the risks of an attack.	L2.NI.C.01 Compare and refine ways in which software developers protect devices and information from unauthorized access.
	Practice: Communicating About Computing				
	6.NI.C.02 Explain the importance of secured websites and describe how encryption works.	7.NI.C.02 Identify and explain methods of encryption used to ensure and secure the transmission of information.	8.NI.C.02 Compare the advantages and disadvantages of methods of encryption to model the secure transmission of information.	L1.NI.C.02 Recommend security measures to address various scenarios based on information security principles.	
	Practice: Developing and Using Abstractions			*Practice: Recognizing and Defining Computational Problems*	
	Additional standards for this subconcept begin in L1.			L1.NI.C.03 Explain trade-offs when selecting and implementing cybersecurity recommendations from multiple perspectives, such as the user, enterprise, and government.	
				Practice: Communicating About Computing	
Data Analysis					
Subconcept	**6th Grade**	**7th Grade**	**8th Grade**	**Level 1 – By the end of 10th Grade**	**Level 2 – By the end of 12th Grade**
Storage	6.DA.S.01 Create multiple representations of the same data.	7.DA.S.01 Create and compare multiple representations of the same data.	8.DA.S.01 Analyze multiple methods of representing the same data and justify the most appropriate method for representing data.	L1.DA.S.01 Convert and compare different bit representations of data types, such as characters, numbers, and images.	*Students will continue to apply the standards from the previous grade levels.*
	Practice: Developing and Using Abstractions				

	Additional standards for this subconcept begin in L1.			L1.DA.S.02 Evaluate the trade-offs in how data is organized and stored digitally. *Practice: Recognizing and Defining Computational Problems*	
Collection, Visualization, & Transformation	6.DA.CVT.01 Collect data using computational tools and transform the data to make it more useful.	7.DA.CVT.01 Collect data using computational tools and transform the data to make it more useful and reliable.	8.DA.CVT.01 Develop, implement, and refine a process that utilizes computational tools to collect and transform data to make it more useful and reliable.	L1.DA.CVT.01 Use tools and techniques to locate, collect, and create visualizations of small and largescale data sets (e.g., paper surveys and online data sets).	L2.DA.CVT.01Use data analysis tools and techniques to identify patterns from complex real-world data.
	Practice: Testing and Refining Computational Artifacts			*Practice: Developing and Using Abstractions*	
	Additional standards for this subconcept begin in L2.				L2.DA.CVT.02 Generate data sets that use a variety of data collection tools and analysis techniques to support a claim and/or communicate information. *Practice: Communicating About Computing*
Inference & Models	6.DA.IM.01 Use data to highlight or propose cause-and-effect relationships, predict outcomes, and communicate ideas.	7.DA.IM.01 Discuss the accuracy of a model representing a system by comparing the model's generated results with observed data from the modeled system.	8.DA.IM.01 Refine computational models based on the data generated by the models.	L1.DA.IM.01 Illustrate and explain the relationships between collected data elements using computational models.	L2.DA.IM.01 Use models and simulations to help plan, conduct, and refine investigations.
	Practice: Developing and Using Abstractions		*Practices: Creating Computational Artifacts, Developing and Using Abstractions*	*Practice: Developing and Using Abstractions*	
Algorithms & Programming					
Subconcept	**6th Grade**	**7th Grade**	**8th Grade**	**Level 1 – By the end of 10th Grade**	**Level 2 – By the end of 12th Grade**

<table>
<tr><td rowspan="6">Algorithms</td><td>6.AP.A.01 Use an existing algorithm in natural language or pseudocode to solve complex problems.</td><td>7.AP.A.01 Select and modify an existing algorithm in natural language or pseudocode to solve complex problems.</td><td>8.AP.A.01 Design algorithms in natural language, flow and control diagrams, comments within code, and/or pseudocode to solve complex problems.</td><td>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.</td><td>L2.AP.A.01 Model and use appropriate terminology to describe how artificial intelligence algorithms drive many software and physical systems (e.g., autonomous robots, pattern recognition, text analysis.)</td></tr>
<tr><td colspan="3">Practice: Developing and Using Abstractions</td><td>Practice: Creating Computational Artifacts</td><td>Practice: Communicating About Computing</td></tr>
<tr><td colspan="4" rowspan="4">Additional Standards for this subconcept begin in L2.</td><td>L2.AP.A.02 Develop an artificial intelligence algorithm to play a game against a human opponent or solve a real-world problem.</td></tr>
<tr><td>Practice: Creating Computational Artifacts</td></tr>
<tr><td>L2.AP.A.03 Critically examine and trace classic algorithms (e.g., selection sort, insertion sort, binary search, linear search).</td></tr>
<tr><td>Practice: Developing and Using Abstractions</td></tr>
<tr><td rowspan="2">Variables</td><td colspan="3" rowspan="2">Standards for this subconcept begin in L1.</td><td>L1.AP.V.01 Demonstrate the use of lists (e.g., arrays) to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.</td><td>L2.AP.V.01 Compare and contrast data structures and their uses (e.g., lists, stacks, queues).</td></tr>
<tr><td colspan="2">Practice: Developing and Using Abstractions</td></tr>
<tr><td>Control</td><td>6.AP.C.01 Develop programs that utilize combinations of repetition, conditionals, and the manipulation of variables representing different data types.</td><td>7.AP.C.01 Develop programs that utilize combinations of repetition, compound conditionals, and the manipulation of variables representing different data types.</td><td>8.AP.C.01 Develop programs that utilize combinations of nested loops, compound conditionals, procedures without parameters, and the manipulation of variables</td><td>L1.AP.C.01 Justify the selection of specific control structures (e.g., sequence, conditionals, repetition, procedures) considering program efficiencies such as readability, performance, and memory usage.</td><td>L2.AP.C.01 Model the execution of repetition (e.g., loops, recursion) of an algorithm illustrating output and changes in values of named variables.</td></tr>
</table>

			representing different data types.		
	Practice: Creating Computational Artifacts			*Practice: Recognizing and Defining Computational Problems*	
Modularity	6.AP.M.01 Decompose problems into parts to facilitate the design, implementation, and review of programs.	7.AP.M.01 Decompose problems into parts to facilitate the design, implementation, and review of increasingly complex programs.	8.AP.M.01 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of complex programs.	L1.AP.M.01 Decompose problems into procedures using systematic analysis and design.	L2.AP.M.01 Construct solutions to problems using student-created components (e.g., procedures, modules, objects).
	Practice: Creating Computational Artifacts			*Practice: Recognizing and Defining Computational Problems*	*Practice: Creating Computational Artifacts*
	Additional standards for this subconcept begin at L1.			L1.AP.M.02 Create computational artifacts by systematically organizing, manipulating and/or processing data.	L2.AP.M.02 Design or redesign a solution to a large-scale computational problem by identifying generalizable patterns.
				Practice: Recognizing and Defining Computational Problems	*Practice(s): Developing and Using Abstractions*
					L2.AP.M.03 Create programming solutions by reusing existing code (e.g., libraries, Application Programming Interface (APIs), code repositories).
					Practice: Creating Computational Artifacts
Program Development	6.AP.PD.01 Seek and incorporate feedback from team members to refine a solution to a problem.	7.AP.PD.01 Seek and incorporate feedback from team members and users to refine a solution to a problem.	8.AP.PD.01 Seek and incorporate feedback from team members and users to refine a solution to a problem that meets the needs of diverse users.	L1.AP.PD.01 Create software by analyzing a problem and/or process, developing and documenting a solution, testing outcomes, and adapting the program for a variety of users.	L2.AP.PD.01 Create software that will provide solutions to a variety of users using the software life cycle process.
	Practices: Collaborating Around Computing, Fostering an Inclusive Computing Culture			*Practice: Communicating About Computing*	*Practice: Creating Computational Artifacts*

	6.AP.PD.02 Incorporate existing code, media, and libraries into original programs and give attribution.	7.AP.PD.02 Incorporate existing code, media, and libraries into original programs of increasing complexity and give attribution.	8.AP.PD.02 Incorporate existing code, media, and libraries into original programs of increasing complexity and give attribution.	L1.AP.PD.02 Evaluate a variety of software licensing schemes (e.g., open source, freeware, commercial) and discuss the advantages and disadvantages of each scheme in software development.	L2.AP.PD.02 Design software in a project team environment using integrated development environments (IDEs), versioning systems, and collaboration systems.
	Practices: Developing and Using Abstractions, Creating Computational Artifacts, Communicating About Computing			*Practice(s): Communicating About and Collaborating Around Computing*	
	6.AP.PD.03 Test and refine programs using teacher provided inputs.	7.AP.PD.03 Test and refine programs using a variety of student created inputs.	8.AP.PD.03 Systematically test and refine programs using a range of student created inputs.	L1.AP.PD.03 While working in a team, develop, test, and refine event-based programs that solve practical problems or allow self expression.	L2.AP.PD.03 Develop programs for multiple computing platforms.
	Practice: Testing and Refining Computational Artifacts				*Practice: Creating Computational Artifacts*
	6.AP.PD.04 Break down tasks and follow an individual timeline when developing a computational artifact.	7.AP.PD.04 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.	8.AP.PD.04 Model effective communication between participants and demonstrate successful collaboration when developing computational artifacts.	L1.AP.PD.04 Using visual aids and documentation, illustrate the design elements and data flow (e.g., flowcharts, pseudocode) of the development of a complex program.	L2.AP.PD.04 Systematically examine code for correctness, usability, readability, efficiency, portability, and scalability through peer review.
	Practice: Collaborating Around Computing			*Practice: Communicating About Computing*	*Practice: Testing and Refining Computational Artifacts*
	6.AP.PD.05 Document text-based programs in order to make them easier to follow, test, and debug.	7.AP.PD.05 Document text-based programs of increasing complexity in order to make them easier to follow, test, and debug.	8.AP.PD.05 Document text-based programs of increasing complexity in order to make them easier to follow, test, and debug.	L1.AP.PD.05 Evaluate and refine computational artifacts to make them more user-friendly, efficient and/or accessible.	L2.AP.PD.05 Develop and use a series of test cases to verify that a program performs according to its design specifications.
	Practice(s): Communicating About Computing			*Practice(s): Testing and Refining Computational Artifacts*	
	Additional standards for this subconcept begin in L2.				L2.AP.PD.06 Explain security issues that might lead to compromised computer programs.

					Practice: Communicating About Computing
					L2.AP.PD.07 Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
					Practice: Creating Computational Artifacts
Impacts of Computing					
Subconcept	**6th Grade**	**7th Grade**	**8th Grade**	**Level 1 – By the end of 10th Grade**	**Level 2 – By the end of 12th Grade**
Culture	6.IC.C.01 Explain how computing impacts people's everyday activities and careers.	7.IC.C.01 Describe the trade-offs associated with computing technologies (e.g., automation), explaining their effects on economies and global societies.	8.IC.C.01 Explore careers related to the field of computer science, and explain how computing impacts innovation in various career fields.	L1.IC.C.01 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.	L2.IC.C.01 Evaluate the beneficial and harmful effects that computational artifacts and innovations have on society.
	Practice(s): Communicating About Computing				
	6.IC.C.02 Identify and discuss the technology proficiencies needed in the classroom and the workplace, and how to meet the needs of diverse users.	7.IC.C.02 Identify real-world problems (e.g., inequities, lack of access) in relation to the distribution of computing resources in a global society.	8.IC.C.02 Evaluate and improve the design of existing technologies to meet the needs of diverse users and increase accessibility and usability.	L1.IC.C.02 Test and refine computational artifacts to reduce bias and equity deficits.	L2.IC.C.02 Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.
	Practice: Fostering an Inclusive Computing Culture				
	Additional standards for this subconcept begin at L1.			L1.IC.C.03 Demonstrate ways a given algorithm can help solve computational problems across disciplines.	L2.IC.C.03 Design and implement a study that evaluates or predicts how creating, testing, and refining computational artifacts has revolutionized an aspect of our culture and how it might evolve (e.g., education, healthcare, art/entertainment, energy).
				Practice: Recognizing and Defining Computational Problems	*Practice: Communicating About Computing*

Social Interactions	6.IC.SI.01 Describe and use safe, appropriate, and responsible practices (i.e., netiquette) when participating in online communities.	7.IC.SI.01 Describe and use safe, appropriate, and responsible practices (i.e., netiquette) when participating in online communities and evaluate how technology can be used to distort, exaggerate, and misrepresent information.	8.IC.SI.01 Describe and use safe, appropriate, and responsible practices (i.e., netiquette) when participating in online communities and understand the impact of not using safe, appropriate, and responsible practices.	L1.IC.SI.01 Demonstrate and debate how computing increases and decreases connectivity and communication among people of various cultures.	
	Practice: Collaborating Around Computing				
	6.IC.SI.02 Individually and collaboratively develop and conduct an online survey that seeks input from a broad audience. Use the survey to evaluate whether it is feasible to solve a problem computationally.	7.IC.SI.02 Individually and collaboratively use advanced tools to design and create online content (e.g., digital portfolio, multimedia, blog, web page).	8.IC.SI.02 Communicate and publish key ideas and details individually or collaboratively in a way that informs, persuades, and/or entertains using a variety of digital tools and media-rich resources.		
	Practices: Collaborating Around Computing, Creating Computational Artifacts				
Safety, Law, & Ethics	6.IC.SLE.01 Differentiate between appropriate and inappropriate content on the Internet, and identify the characteristics of unethical and illegal online behavior.	7.IC.SLE.01 Model the connection between the longevity of data on the Internet, personal online identity, and personal privacy.	8.IC.SLE.01 Discuss the social impacts and ethical considerations associated with cybersecurity, including the positive and malicious purposes of hacking.	L1.IC.SLE.01 Describe the beneficial and harmful effects that intellectual property laws can have on innovation.	L2.IC.SLE.01 Debate laws and regulations that impact the development and use of software.
	Practice: Communicating About Computing				*Practices: Recognizing and Defining Computational Problems, Communicating About Computing*
	Additional standards for this subconcept begin in L1.			L1.IC.SLE.02 Describe and discuss the privacy concerns related to the large-scale collection and analysis of information about individuals (e.g., how websites collect and uses data) that may not be evident to users.	
				Practice: Communicating About Computing	

		L1.IC.SLE.03 Evaluate the social and economic consequences of how law and ethics interact with digital aspects of privacy, data, property, information, and identity. *Practice: Communicating About Computing*	