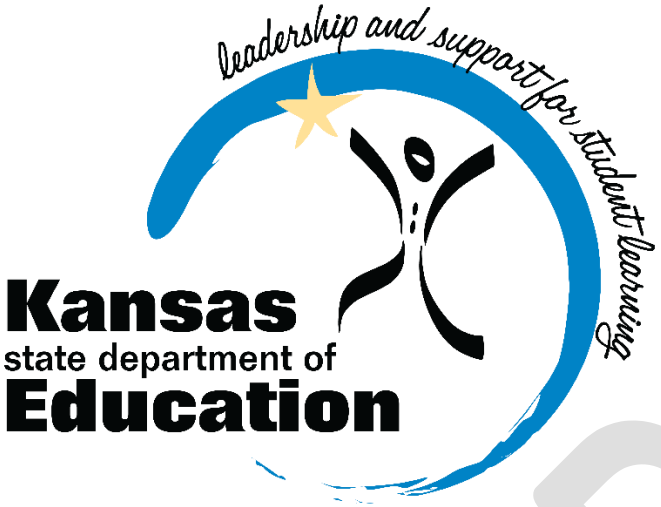

Kansas Computer Science Standards

Grades P-12

Adopted (DATE)

About the Kansas P-12 Computer Science Model Standards

To be well-educated citizens in a computing-intensive world and to be prepared for careers in the 21st century, our students must have a clear understanding of the principles and practices of computer science. The Kansas P-12 Computer Science Model Standards delineate a core set of learning objectives designed to provide the foundation for a complete computer science curriculum and its implementation at the P–12 level. To this end, the Standards:

- Introduce the fundamental concepts of computer science to all students, beginning at the primary school level.
- Develop the practices of computational thinking in a sequential progression from pre-kindergarten through high school.
- 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 work force or college.
- Increase the availability of rigorous computer science for all students, especially those who are members of underrepresented groups.

The standards have been written by educators to be coherent and comprehensible to teachers, administrators, and policy makers. Grades P-5, middle grades, and secondary L1 are the computer science standards for **all students**. The secondary L2 standards are intended for students who wish to pursue the study of computer science beyond what is expected of all students (specialty or elective courses).

Connection to the K-12 Computer Science Framework and CSTA Standards

The K–12 Computer Science Framework (k12cs.org) provides overarching, high-level guidance per grade bands, while the standards provide detailed, measurable student performance expectations. The Framework, and the subsequent Computer Science Teachers Association (CSTA) standards document, were considered as primary inputs for the Kansas standards development process.

Concepts

1. Computing Systems
2. Networks and the Internet
3. Data Analysis
4. Algorithms and Programming
5. Impacts of Computing

Practices

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

Kansas Computer Science Committee

Information about standards committee formation was shared with the education community via KSDE listservs, meetings, and the State Board of Education. A registration site was developed with the purpose of obtaining nominations for the standards development committees. Individuals could either self-nominate or could recommend someone. The registration site asked for name, address, email, board district, job title, gender, race, education level, committee group interest, and years of work experience. KSDE staff were asked to ensure that committee members for the standards committees consisted of diversity of gender, race, ethnicity, and education level (K-12 and post-secondary). Special care was taken to ensure that every state board district was represented.

In addition to the committee members the computer science standards committee had a "Representative" ad-hoc group which was comprised of postsecondary, business/community, and military representatives. These individuals were interested in the standards review process and their role was to participate in the discussions and provide feedback.

Writing Subcommittee	Review Subcommittee	Representative Group
Chris Holborn, USD 475	Amy Benz, USD 320	Robert Burcham, Business & Industry
Gwen Lehman, USD 495	Tyler Bruce, USD 483	Steven Case, PhD, University of Kansas
Laura Leis, USD 400	Pam Collinge, USD 389	Charmine Chambers, KBOR
Craig Miller, USD 475	Ross Davis, USD 508	Anna Hennes, Business & Industry
Shane Munro, USD 259	David Dennis, USD 259	Chris Issacson, Business & Industry
Matthew Peak, USD 503	Barbra Gonzales, USD 233	David Kaercher, Business & Industry
Kristy Randel, USD 253	Matthew Lewis, USD 259	Meg Knauth, Business & Industry
Gary Richmond, USD 464	Brittney Quelch, USD 475	Jason Knobbe (COL), Military
Bryan Salsgiver, USD 229	Brenda Thompson, USD 373	Alan Lowden, Business & Industry
Samuel Simmons, Sr. USD 500	Lisa Whallon, USD 233	Ryan Weber, Business & Industry
Steven Stoffregen, USD 465	Tyler Wolf, USD 348	Bruce Wellman, NGSS
Jill Thompson, USD 264	Kelley Wyatt, USD 464	
Travis True, USD 501		
Josh Weese, PhD, Kansas State University		
Chris Wyant, Wichita State University		

Grade PK (Pre-Kindergarten)

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
PK.CS.D.01	With guidance, demonstrate how to operate a computing device. *People use computing devices to perform a variety of tasks accurately and quickly. With guidance, students should be able to select the appropriate app/program to use for tasks they are required to complete. For example, if students are asked to draw a picture, they should be able to open and use a drawing app/program to complete this task.*	Devices	7. Communicating about computing
PK.CS.HS.01	With guidance, use appropriate terminology to locate and identify common computing devices and components in a variety of environments (e.g. turn on, navigate, open/close programs/apps). *A computing system is composed of hardware and software. Hardware consists of physical components. With guidance, students should be able to identify and describe the function of external hardware, such as desktop computers, laptop computers, tablet devices, monitors, keyboards, mice, and printers.*	Hardware and Software	7. Communicating about computing
PK.CS.HS.02	With guidance, correctly use software that controls computing devices (e.g. e.g. programs, browsers, websites, and applications). *Computer software and apps are programmed and installed on hard drives on various devices utilized by every end user. Software provides code for the programs to compute properly for the created operation. Software apps and programs interact with one another to provide an intended outcome or output. With guidance, students should be able to open, use, and close varying programs, apps, or software.*	Hardware and Software	1. Fostering an inclusive computing culture
PK.CS.IO.01	With guidance, identify and apply basic input/output skills. • Input (keyboarding, mouse, touchscreen, voice, camera, interactive board) • Output (monitor, screen, printer, audio).	Input and Output	7. Communicating about computing

	Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice input, touchpad, touchscreen, mouse, keyboarding (Keyboarding - practice locating space bar, enter key, and developmentally appropriate letters.) Output devices are how a computer displays information, which includes the screen, monitor, speaker, or printer.		
PK.CS.T.01	Recognize that computing systems might not work as expected and with guidance can identify simple hardware or software problems (e.g. volume turned down on headphones, monitor turned off). *Problems with computing systems have different causes. Students at this level do not need to understand those causes, but they should be able to communicate a problem (e.g., when an app or program is not working as expected, a device will not turn on, the sound does not work, etc.). Ideally, students would be able to use simple troubleshooting strategies, including turning on and off the device, turning on speakers, adjusting volume, or plugging in headphones. These are, however, not specified in the standard, because these problems may not occur.*	Troubleshooting	6. Testing and refining computational artifacts 7. Communicating about computing

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
PK.NI.NCO.01	Recognize that computing devices are connected via wired or wireless networks so that they can communicate with each other. *Networking and interconnectivity of computing devices are essential in today's society. Through wi-fi, bluetooth, or hard line ethernet connections, the ability of information to be shared with an organized, secure and reliable system, is an integrated range of platforms which uses various software and hardware. Students should have an awareness the device is connected to another device.*	Network Communication & Organization	7. Communicating about computing
PK.NI.C.01	Recognize that passwords are private and should be kept secret. *Learning to protect one's device or information from unwanted use by others is an essential first step in learning about cybersecurity.*	Cybersecurity	7. Communicating about computing

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
PK.DA.S.01	Know that the computing device can save information as data that can be searched, modified, and saved or deleted (e.g. save photos, files, or videos). *All information stored and processed by a computing device is referred to as data. Data can be images, text documents, audio files, software programs or apps, video files, etc. As students use software to complete tasks on a computing device, they will be manipulating data. Students should be aware that information can be found or searched on a device.*	Storage	4. Developing and using abstractions
PK.DA.C.01	Students understand that data about themselves and the world around them is collected, used, and organized in a meaningful way. *The collection and use of data about the world around them is a routine part of life and influences how people live.*	Collection	4. Developing and using abstractions 7. Communicating about computing
PK.DA.CVT.01	Students represent collected data in a visual way. (e.g. charts, graphs, tables). *Data can be used to make inferences or predictions about the world. Students could analyze a graph or pie chart of the colors in a bag of candy or identify the patterns for which colors are most and least represented, and then make a prediction as to which colors will have most and least in a new bag of candy. Students can explore bar graphs or line graphs to analyze what has more or less. This can be done without a computing device - paper, interactive board, chart paper, class graph, etc.*	Visualization & Transformation	7. Communicating about computing
PK.DA.IM.01	Students look for patterns in data, make predictions, and make a model (e.g. make predictions on weather data, butterfly life cycle, etc.) and present in a picture graph or pattern. *Data can be represented in models to portray results and to assist in identifying patterns in the world around us. This type of data is represented in a more visual way outside of lines, bars, and charts. This would include life cycles, weather maps, and processes. Students will*	Inference and Models	4. Developing and using abstractions

	show data in a pattern. With guidance, students will show what would be next in a basic pattern, or what might be missing from a pattern. This could be a color pattern, number pattern, animal pattern, etc. It can be as basic as ABAB, or ABBABB.		

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
PK.AP.A.01	With guidance, construct and execute algorithms (set of step-by-step instructions) that includes sequencing and simple loops to accomplish a task, with or without a computing device (e.g. verbally, kinesthetically, with robot devices or a programming language, block coding). *Algorithmic thinking is the ability to define clear steps to solve a problem. A process to complete a task (such as the steps to tie your shoes), and recipes are examples of algorithms. Expose students to the term algorithm as they are sequencing events or processes like getting ready for school in the morning.*	Algorithms	4. Developing and using abstractions
PK.AP.V.01	With guidance, understand that numbers represent different types of data using numbers or other symbols (e.g. thumbs up/thumbs down for yes/no color by number, arrows for direction, encoding/decoding a word using numbers or pictographs). *Information in the real world can be represented in computer programs. Students could use thumbs up/down as representations of yes/no, use arrows when writing algorithms to represent direction, or encode and decode words using numbers, pictographs, or other symbols to represent letters or words.*	Variables	4. Developing and using abstractions
PK.AP.C.01	With guidance, create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing events and simple loops (e.g. emphasizing beginning, middle, and end; collaborative programming). *Programming is used as a tool to create products that reflect a wide range of interests. Control structures specify the order in which instructions are executed within a program. Emphasize the sequence of events, such as left right, up, down. Get from one point to another on a*	Control	5. Creating computational artifacts

	map. Explore basic robots that use arrows for direction, or search for lessons on CS unplugged fundamentals.		
PK.AP.M.01	With guidance, decompose (break down) a larger problem into smaller subproblems. *Decomposition is the act of breaking down tasks into simpler tasks. Students could break down the steps needed to make breakfast, get ready for school, to move a character across the screen. This can be done with or without a computing device.*	Modularity	3. Recognizing and defining computational problems
PK.AP.PD.01	Create a design document to illustrate thoughts, ideas, and stories in a sequential manner. *Creating a design document for what a program will do clarifies the steps that will be needed to create a program and can be used to check if a program is correct. Students could create a planning document such as a story map to illustrate what their program will do.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
PK.AP.PD.02	Recognize that digital items can be owned and that proper credit needs to be given (e.g. using code, music, pictures). *Using computers comes with a level of responsibility. Students should recognize that artifacts were created by others, such as pictures, music, and code.*	Program Development	7. Communicating about computing
PK.AP.PD.03	With guidance, construct, execute, and debug (identify and fix) algorithms using a programming language and or an unplugged activity that includes sequencing (e.g. use block based programming). *Algorithms or programs may not always work correctly. With guidance, students should be able to use various strategies, such as changing the sequence of the steps, following the algorithm in a step-by-step manner, or trial and error to fix problems in algorithms and programs.*	Program Development	6. Testing and refining computational artifacts
PK.AP.PD.04	With guidance, use correct terminology in the development of an algorithm to solve a simple problem (e.g. beginning, middle, end). *With guidance, students should be able to talk or write about the goals and expected outcomes of the programs they create and the choices*	Program Development	7. Communicating about computing

	that they made when creating programs. This could be done through discussions with the teacher or class.		

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
PK.IC.C.01	Understand different ways in which types of technologies are used in your daily life. *In the past, if students wanted to read about a topic, they needed access to a library to find a book about it. Today, young students can view and read information on the Internet about a topic or they can download e-books about it directly to a device. Such information may be available in more than one language and could be read to a student, allowing for great accessibility. Students should develop an awareness in describing various ways technology can impact their world. (e.g. checking out at a store, buying lunch, using an iPhone or Android device to call in an emergency, or learning through video sharing).*	Culture	7. Communicating about computing
PK.IC.SI.01	With guidance understand what would be appropriate while participating in an online environment. (Digital Citizenship - focus on Digital Literacy). *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Communicate to students the importance of being safe online by only using sites approved by an adult. Encourage students to tell an adult if they feel uncomfortable or see something they feel is not appropriate. The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Digital citizenship is described with nine categories, however PreK-2 will focus on 4 of these: Digital Literacy (the ability to use new technology quickly and appropriately), Digital Etiquette (appropriate conduct), Digital Rights and Responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online), and Digital Health and Wellness (caring for your physical and psychological well-being online).*	Social Interactions	2. Collaborating around computing

PK.IC.H.01	Understand that computing technology has changed and improved the way people live, work, and interact. *As computers become interconnected in each aspect of society, more powerful, and students become more reliant on them, students will engage in discussions about how they have evolved since their parents were in school and relate the newest devices they have at home.*	History	7. Communicating about computing
PK.IC.SLE.01	With guidance understand responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software. *People use computing technology in ways that can help or hurt themselves or others. Expose students to sharing devices and leaving the device ready for the next user (closing programs, etc.). Explain how passwords or login methods are used and why we protect devices with these.*	Safety, Law, & Ethics	2. Collaborating around computing
PK.IC.CP.01	Discuss the fact that a wide range of jobs require knowledge or use of computer science. *Within the inevitable interwoven fabric of society's reliance and innovative machines, students will required to have basic assumable skills when entering the workforce. Students should be able with guidance, picture digital computing devices and word usage necessary to create a modernized mode of everyday activities in the technological age. An example would be for students to list how a bus driver can use GPS, safety features, and indicators to provide safe travel to school.*	Community Partnerships	7. Communicating about computing

Grade K (Kindergarten)

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
K.CS.D.01	Demonstrate how to operate a variety of computing devices. *People use computing devices to perform a variety of tasks accurately and quickly. After Instruction, students should be able to select the appropriate app/program to use for tasks they are required to complete, then power down or log off. For example, if students are asked to draw a picture, they should be able to open and use a drawing app/program to complete this task.*	Devices	7. Communicating about computing
K.CS.HS.01	Use appropriate terminology to locate and identify common computing devices and components in a variety of environments (e.g. turn on, navigate, open/close programs/apps). *A computing system is composed of hardware and software. Hardware consists of physical components. After instruction, students should be able to identify and describe the function of external hardware, such as desktop computers, laptop computers, tablet devices, monitors, keyboards, mice, and printers.*	Hardware and Software	7. Communicating about computing
K.CS.HS.02	Identify and use software that controls computing devices (e.g. programs, browsers, websites, and applications). *Computer software and apps are programmed and installed on hard drives on various devices utilized by every end user. Software provides code for the programs to compute properly for the created operation. Software apps and programs interact with one another to provide an intended outcome or output. With guidance, students should be able to associate the icon with the appropriate program/application and its use, then open, use, and close programs, apps, or software. This could include, but not limited to, district purchased client-based reading or math program software, apps for a specific learning method, or accessing a browser to navigate web based programs.*	Hardware and Software	1. Fostering an inclusive computing culture
K.CS.IO.01	Identify and apply basic input/output skills. • Input (keyboarding, mouse, touchscreen, voice, camera, robotics, interactive board)	Input and Output	7. Communicating about computing

	• Output (monitor, screen, printer, robotics, audio). *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice input, touchpad, touchscreen, mouse, keyboarding (Keyboarding - practice locating space bar, enter key, and developmentally appropriate letters. Students should understand the left hand is used for the left side of the keyboard, and the right hand is used on the right side. This includes the understanding the general layout of the keys including developmentally appropriate number recognition.) Output devices are how a computer displays information, which includes the screen, monitor, speaker, or printer.*		
K.CS.T.01	Recognize that computing systems might not work as expected and use accurate terminology to identify simple hardware or software problems (e.g. volume turned down on headphones, monitor turned off). *Problems with computing systems have different causes. Students at this level do not need to understand those causes, but they should be able to communicate a problem with accurate terminology (e.g., when an app or program is not working as expected, a device will not turn on, the sound does not work, etc.). Ideally, students would be able to use simple troubleshooting strategies, including turning a device off and on to reboot it, closing and reopening an app, turning on speakers, or plugging in headphones. These are, however, not specified in the standard, because these problems may not occur.*	Troubleshooting	6. Testing and refining computational artifacts 7. Communicating about computing

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
K.NI.NCO.01	Recognize and use computing devices to connect with people or other devices using a network to communicate, access, and share information as a class (e.g. the internet, video conferencing, email, file transfer). *Networking and interconnectivity of computing devices are essential in today's society. Through wi-fi, bluetooth, or hard line ethernet connections, the ability of information to be shared with an organized, secure and reliable system, is an integrated range of platforms which*	Network Communication & Organization	7. Communicating about computing

	uses various software and hardware. Students should understand whether information is being sent to the program or device. (e.g., the teacher laptop is being connected to the LCD projector, or if the wi-fi or internet connection is active.		
K.NI.C.01	Use a form of secure access to protect private information and discuss the effects of password misuse (e.g. logging into a device, educational websites, authentication, thumbprint recognition). *Learning to protect one's device or information from unwanted use by others is an essential first step in learning about cybersecurity. Students should appropriately use and protect the passwords they are required to use.*	Cybersecurity	7. Communicating about computing

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
K.DA.S.01	With guidance, demonstrate that computing devices can save information as data that can be searched, modified, and saved or deleted (e.g. save photos, files, or videos). *All information stored and processed by a computing device is referred to as data. Data can be images, text documents, audio files, software programs or apps, video files, etc. As students use software to complete tasks on a computing device, they will be manipulating data. With guidance, students will search, save, or delete data. This can be a web search, save and retrieve a photo, take a screenshot, or saving or printing their creations.*	Storage	4. Developing and using abstractions
K.DA.C.01	Students will learn how data about themselves and the world around them is collected, used, and organized in a meaningful way. *The collection and use of data about the world around them is a routine part of life and influences how people live. Students could collect data on the weather, such as sunny days versus rainy days, the temperature at the beginning of the school day and end of the school day, or the inches of rain over the course of a storm.*	Collection	4. Developing and using abstractions 7. Communicating about computing
K.DA.CVT.01	Students represent collected data in a visual way through a computing device (e.g. charts, graphs, tables).	Visualization & Transformation	7. Communicating about computing

	Data can be used to make inferences or predictions about the world. Students could analyze a graph or pie chart of the colors in a bag of candy or the averages for colors in multiple bags of candy, identify the patterns for which colors are most and least represented, and then make a prediction as to which colors will have most and least in a new bag of candy. With guidance, students could create charts or graphs in spreadsheet applications, web based programs, or visually in digital drawings to portray data collected. This data could include types of pets, number of siblings, shoe size, etc. This could be done with an interactive board, tablets, or computer.		
K.DA.IM.01	Students look for patterns in data, make predictions, make a model, and draw conclusions (e.g. make predictions on weather data, butterfly life cycle, etc.) and present in a picture graph or pattern. *Data can be represented in models to portray results and to assist in identifying patterns in the world around us. This type of data is represented in a more visual way outside of lines, bars, and charts. This would include life cycles, weather maps, and processes. Students will show data in a pattern. Students will create models to show data which could include pictographs of favorite cookie, fruit, sport, or models also include Students will show what would be next in a pattern, or what might be missing from a pattern. This could be a color pattern, number pattern, animal pattern, etc. It can be as basic as ABAB, or ABBABB.*	Inference and Models	4. Developing and using abstractions

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
K.AP.A.01	Construct and execute algorithms (set of step-by-step instructions) that includes sequencing and simple loops to accomplish a task, both independently, collaboratively, with or without a computing device (e.g. verbally, kinesthetically, with robot devices or a programming language, block coding). *Algorithmic thinking is the ability to define clear steps to solve a problem. A process to complete a task (such as the steps to tie your*	Algorithms	4. Developing and using abstractions

	shoes), and recipes are examples of algorithms. Expose students to the term algorithm as they are sequencing events or processes like getting ready for school in the morning. Students should create algorithms (specific steps) to accomplish a task.		
K.AP.V.01	With guidance, recognize that numbers represent different types of data using numbers or other symbols (e.g. thumbs up/thumbs down for yes/no color by number, arrows for direction, encoding/decoding a word using numbers or pictographs). *Information in the real world can be represented in computer programs. Students could use thumbs up/down as representations of yes/no, use arrows when writing algorithms to represent direction, use emojis that represent emotion, or encode and decode words using numbers, pictographs, or other symbols to represent letters or words.*	Variables	4. Developing and using abstractions
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 events and simple loops (e.g. emphasizing beginning, middle, and end; collaborative programming). *Programming is used as a tool to create products that reflect a wide range of interests. Control structures specify the order in which instructions are executed within a program. Emphasize the sequence of events, such as left right, up, down. Get from one point to another on a map. Explore basic robots that use arrows for direction, or search for lessons on CS unplugged fundamentals.*	Control	5. Creating computational artifacts
K.AP.M.01	With guidance, decompose (break down) a larger problem into smaller subproblems or combine simple tasks to make something more complex. *Decomposition is the act of breaking down tasks into simpler tasks. Students could break down the steps needed to make breakfast, get ready for school, to move a character across the screen. Combining tasks could include being given objects to construct sub parts that go together to make a more complex creation (e.g., building structures out*	Modularity	3. Recognizing and defining computational problems

	of Legos, then combining them into a town or community). This can be done with or without a computing device.		
K.AP.PD.01	Create a design document to illustrate thoughts, ideas and stories in a sequential manner (e.g. storyboard, mindmap, sequential graphic organizer). *Creating a design document for what a program will do clarifies the steps that will be needed to create a program and can be used to check if a program is correct. Students could create a planning document, such as a story map, a storyboard, or a sequential graphic organizer, to illustrate what their program will do. Students at this stage may complete the planning process with help from their teachers.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
K.AP.PD.02	With guidance, give credit to ideas, creations, and solutions of others while developing algorithms (e.g. using code, music, pictures). *Using computers comes with a level of responsibility. With guidance, students should credit artifacts that were created by others, such as pictures, music, and code. Credit could be given orally, if presenting their work to the class, or in writing or orally, if sharing work on a class blog or website. Proper attribution at this stage does not require a formal citation, such as in a bibliography or works cited document.*	Program Development	7. Communicating about computing
K.AP.PD.03	With guidance, independently or collaboratively construct, execute, and debug (identify and fix) algorithms using a programming language and or an unplugged activity that includes sequencing (e.g. use block based programming). *Algorithms or programs may not always work correctly. With guidance, students should be able to use various strategies, such as changing the sequence of the steps, following the algorithm in a step-by-step manner, or trial and error to fix problems in algorithms and programs independently or collaboratively.*	Program Development	6. Testing and refining computational artifacts
K.AP.PD.04	Use correct terminology in the development of an algorithm to solve a simple problem (e.g. beginning, middle, end). *At this stage, students should be able to talk or write about the goals and expected outcomes of the programs they create and the choices*	Program Development	7. Communicating about computing

	that they made when creating programs. This could be done using coding journals, discussions with a teacher, or class presentations.		

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
K.IC.C.01	Understand different ways in which types of technologies are used in your daily life. *In the past, if students wanted to read about a topic, they needed access to a library to find a book about it. Today, students can view and read information on the Internet about a topic or they can download e-books about it directly to a device. Such information may be available in more than one language and could be read to a student, allowing for great accessibility. Students should be able to analyze where and when various ways technology can be used. (e.g., checking out at a store, buying lunch, using an iPhone or Android device to call in an emergency, or learning through video sharing).*	Culture	7. Communicating about computing
K.IC.SI.01	With guidance identify appropriate manners while participating in an online environment. (Digital Citizenship - focus on Digital Literacy and Digital Etiquette) *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Students practice online safety by only using sites approved by an adult. Encourage students to tell an adult if they feel uncomfortable or see something they feel is not appropriate. Make students aware of the privacy of the digital resources being used in the classroom and who sees what is being posted (social media - the teacher posting class photos, students posting to online platforms such as SeeSaw, data from testing sites such as iStation and Lexia.) Digital citizenship is described with nine categories, however PreK-2 will focus on 4 of these: Digital Literacy (the ability to use new technology quickly and appropriately), Digital Etiquette (appropriate conduct), Digital Rights and Responsibilities (knowing your rights to free speech and privacy, but*	Social Interactions	2. Collaborating around computing

	handling it responsibly online), and Digital Health and Wellness (caring for your physical and psychological well-being online).		
K.IC.H.01	Discuss examples of how computing technology has changed and improved the way people live, work, and interact. *As computers become interconnected in each aspect of society, more powerful, and students become more reliant on them, students should be able describe the number of times computers or devices are accessed each day by teachers or peers in class and discuss what life would be like without them.*	History	7. Communicating about computing
K.IC.SLE.01	Practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software. *People use computing technology in ways that can help or hurt themselves or others. Teach students about sharing devices and leaving the device ready for the next user (closing programs, logging out, etc.) Use passwords or other log in methods, learning why we protect devices and programs (such as online assessment) with these. Students should understand they should never post as another person (blogs, SeeSaw, etc.).*	Safety, Law, & Ethics	2. Collaborating around computing
K.IC.CP.01	Understand that a wide range of jobs require knowledge or use of computer science. *Within the inevitable interwoven fabric of society's reliance and innovative machines, students will required to have basic assumable skills when entering the workforce. Students should be able to identify after initial instruction what digital computing devices and languages are necessary to create a modernized mode of everyday activities in the technological age. An example would be for students to list how a bus driver can use GPS, safety features, and indicators to provide safe travel to school.*	Community Partnerships	7. Communicating about computing

First Grade

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
1.CS.D.01	With guidance, select and use a computing device to perform a variety of tasks for an intended outcome. *People use computing devices to perform a variety of tasks accurately and quickly. Students should be able to select the appropriate app/program to use for tasks they are required to complete, then log off or power down. For example, if students are asked to draw a picture, they should be able to open and use a drawing app/program to complete this task, or if they are asked to create a presentation, they should be able to open and use presentation software. In addition, with teacher guidance, students should be aware that different software has the same primary functionality (e.g. Keynote, PowerPoint, Google Slides).*	Devices	7. Communicating about computing
1.CS.HS.01	Use appropriate terminology in identifying and describing the function of common computing devices and components (e.g., use an app to draw on the screen, use software to write a story or control robots). *A computing system is composed of hardware and software. Hardware consists of physical components. Software provides a computer a set of instructions to follow. Students should be able to identify and describe the function of software and hardware such as interactive boards, touch screen devices, and robotics.*	Hardware and Software	7. Communicating about computing
1.CS.HS.02	With guidance select and use appropriate software/apps for an intended outcome (e.g., programs, browsers, websites, and applications). *Computer software and apps are programmed and installed on hard drives on various devices utilized by every end user. Software provides code for the programs to compute properly for the created operation. Software apps and programs interact with one another to provide an intended outcome or output. Students should be able to identify the application or program required for a desired activity. This could include, but not limited to, district purchased client-based reading or math program software, apps for a specific learning methods Reading Eggs,*	Hardware and Software	1. Fostering an inclusive computing culture

	iMovie, Google Apps, Seesaw, or accessing a browser to navigate web based programs.		
1.CS.IO.01	Understand and apply basic input/output skills. • Input (keyboarding, mouse, touchscreen, voice, camera, robotics, interactive board) • Output (monitor, screen, printer, 3D printer, robotics, audio) *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice input, touchpad, touchscreen, mouse, keyboarding (Keyboarding - practice locating space bar, enter key, and developmentally appropriate letters. Students should understand the left hand is used for the left side of the keyboard, and the right hand is used on the right side. This includes the understanding that the keyboard is not in alphabetical order and the general layout of the keys including the location of numbers and basic punctuation.) Output devices are how a computer displays information. Student should understand the use of output devices such as audio, video, screen display, robotics, and printers.*	Input and Output	7. Communicating about computing
1.CS.T.01	Identify and describe basic hardware and software problems using accurate terminology (app or program is not working as expected, no sound is coming from the device, caps lock turned on, wi-fi not working). *Problems with computing systems have different causes. Students at this level will start to understand those causes, communicate the problem with accurate terminology, and seek solutions to that problem (e.g., when an app or program is not working as expected, a device will not turn on, the sound does not work, etc.). Ideally, students would be able to use simple troubleshooting strategies, including turning a device off and on to reboot it, closing and reopening an app, checking wi-fi, turning on speakers, or plugging in headphones. These are, however, not specified in the standard, because these problems may not occur.*	Troubleshooting	6. Testing and refining computational artifacts 7. Communicating about computing

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
1.NI.NCO.01	Recognize that by connecting computing devices together they can share information using a network (e.g. wired or wireless network). *Networking and interconnectivity of computing devices are essential in today's society. Through wi-fi, bluetooth, or hard line ethernet connections, the ability of information to be shared with an organized, secure and reliable system, is an integrated range of platforms which uses various software and hardware. Students should be able to identify whether information is being sent to the program or device. (e.g., the teacher laptop is being connected to the LCD projector, or if how a bluetooth speaker connection is active.)*	Network Communication & Organization	7. Communicating about computing
1.NI.C.01	Identify what authentication methods (passwords) are; explain why they are not shared; and discuss what makes a password strong. Independently, use passwords to access technological devices, apps, etc. *Learning to protect one's device or information from unwanted use by others is an essential first step in learning about cybersecurity. Students are not required to use multiple strong passwords. They should appropriately use and protect the passwords they are required to use.*	Cybersecurity	7. Communicating about computing

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
1.DA.S.01	With guidance locate, open, modify, delete and save an existing file, use appropriate file-naming conventions, and recognize that the file exists within an organizational structure (drive, folder, file). *All information stored and processed by a computing device is referred to as data. Data can be images, text documents, audio files, software programs or apps, video files, etc. As students use software to complete tasks on a computing device, they will be manipulating data. With guidance, students will search for or retrieve files by name, or organize files. This could include taking photos, opening, and deleting them,*	Storage	4. Developing and using abstractions

	organizing files or photos into folders on a desktop or on an operating system, and learning to name and save a file before exiting.		
1.DA.C.01	With guidance, collect data and present it two different ways (chart or graph). *The collection and use of data about the world around them is a routine part of life and influences how people live. Students could collect data on the weather, such as sunny days versus rainy days, the temperature at the beginning of the school day and end of the school day, or the inches of rain over the course of a storm. Students could count the number of pieces of each color of candy in a bag of candy, such as Skittles or M&Ms. Students could create surveys of things that interest them, such as favorite foods, pets, or TV shows, and collect answers to their surveys from their peers and others. The data collected could then be organized into two visualizations, such as a bar graph or pie chart.*	Collection	4. Developing and using abstractions 7. Communicating about computing
1.DA.CVT.01	With guidance, identify and interpret data from a chart or graph (visualization) in order to make a prediction, with or without a computing device. *Data can be used to make inferences or predictions about the world. Students could analyze a graph or pie chart of the colors in a bag of candy or the averages for colors in multiple bags of candy, identify the patterns for which colors are most and least represented, and then make a prediction as to which colors will have most and least in a new bag of candy. Students could create and analyze charts or graphs in spreadsheet applications, web based programs, or visually in digital drawings to portray data collected. They could create and analyze graphs of temperatures taken at the beginning of the school day and end of the school day, identify the patterns of when temperatures rise and fall, and predict if they think the temperature will rise or fall at a particular time of the day, based on the pattern observed. The focus is making predictions based on data.*	Visualization & Transformation	7. Communicating about computing
1.DA.IM.01	Create a model of an object or process in order to identify patterns and essential elements. (e.g. water table, butterfly life cycle, seasonal weather patterns).	Inference and Models	4. Developing and using abstractions

	Data can be represented in models to portray results and to assist in identifying patterns in the world around us. This type of data is represented in a more visual way outside of lines, bars, and charts. This would include life cycles, weather maps, and processes such as the engineering design process. Students will create models either physically (paper, clay, etc.) or digitally using photos, text and shapes with the intent of understanding patterns and essential steps and information.		

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
1.AP.A.01	With guidance, model daily processes and follow algorithms (sets of step-by-step instructions) for complete tasks verbally, kinesthetically, with robot devices, or a programing language. *Algorithmic thinking is the ability to define clear steps to solve a problem. Composition is the combination of smaller tasks into more complex tasks. With guidance, students should be able to create and follow algorithms for making simple foods, brushing their teeth, getting ready for school, participating in clean-up time or programming a robotic device to follow a preset path.*	Algorithms	4. Developing and using abstractions
1.AP.V.01	With guidance, model the way that programs store and manipulate data by using numbers or other symbols to represent information (e.g. thumbs up/thumbs down for yes/no, use arrows when writing algorithms to represent direction, or encode and decode words using numbers, pictographs, or other symbols to represent letters or words). *Information in the real world can be represented in computer programs. Students could use thumbs up/down as representations of yes/no, use arrows when writing algorithms to represent direction, use emojis that represent emotion, or use common icons and symbols to perform an action (play is a triangle, save button, share button, etc.).*	Variables	4. Developing and using abstractions
1.AP.C.01	With guidance, independently, or collaboratively construct algorithms (sets of step-by-step instructions) to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing and repetition, to express ideas or address a problem.	Control	5. Creating computational artifacts

	Programming is used as a tool to create products that reflect a wide range of interests. Control structures specify the order in which instructions are executed within a program. Emphasize the sequence of events, such as left right, up, down. Get from one point to another on a map. Have students develop the steps and have others follow those steps. Search lessons for CS Unplugged, or CS fundamentals.		
1.AP.M.01	With guidance, decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. *Decomposition is the act of breaking down tasks into simpler tasks. Students could break down the steps needed to make a peanut butter and jelly sandwich, to brush their teeth, to draw a shape, to move a character across the screen, or to solve a level of a coding app.*	Modularity	3. Recognizing and defining computational problems
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). *Creating a plan for what a program will do clarifies the steps that will be needed to create a program and can be used to check if a program is correct. Students could create a planning document, such as a story map, a storyboard, or a sequential graphic organizer, to illustrate what their program will do. Students at this stage may complete the planning process by themselves, or with help from their teachers.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
1.AP.PD.02	Independently or with guidance give credit to ideas, creations and solutions of others while writing and/or developing programs. *Using computers comes with a level of responsibility. Students should credit artifacts that were created by others, such as pictures, music, and code. Credit could be given orally, if presenting their work to the class, or in writing or orally, if sharing work on a class blog or website. Proper attribution at this stage does not require a formal citation, such as in a bibliography or works cited document.*	Program Development	7. Communicating about computing
1.AP.PD.03	With guidance, independently, or collaboratively construct, execute, and debug (identify and fix) programs using a programming language and/or unplugged activity that includes sequencing and repetition.	Program Development	6. Testing and refining

	Algorithms or programs may not always work correctly. Students should be able to use various strategies, such as changing the sequence of the steps, following the algorithm in a step-by-step manner, or trial and error to fix problems in algorithms and programs.		computational artifacts
1.AP.PD.04	Use correct terminology (first, second, third) and explain the choices made in the development or an algorithm to solve a simple problem. *At this stage, students should be able to talk or write about the goals and expected outcomes of the programs they create and the choices that they made when creating programs. This could be done using coding journals, discussions with a teacher, class presentations, or blogs.*	Program Development	7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
1.IC.C.01	Identify how people use different types of technologies in their daily work and personal lives. *Computing technology has changed the way people live and work. In the past, if students wanted to read about a topic, they needed access to a library to find a book about it. Today, students will be able to view and read information on the Internet about a topic or they can download e-books about it directly to a device. Such information may be available in more than one language and could be read to a student, allowing for great accessibility. In personal lives, they are encouraged to engage in computing in a positive learning and encouraging manner.*	Culture	7. Communicating about computing
1.IC.SI.01	With guidance, identify appropriate and inappropriate behavior. Act responsibly while participating in an online community and know how to report concerns. (Digital Citizenship - review Digital Literacy, but focus on Digital Etiquette and Rights and Responsibilities) *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Students practice online safety by only using sites approved by an adult. Encourage students to tell an adult if they feel*	Social Interactions	2. Collaborating around computing

	uncomfortable or see something they feel is not appropriate. Make students aware of the privacy of the digital resources being used in the classroom and who sees what is being posted (social media - the teacher posting class photos, students posting to online platforms such as SeeSaw, data from testing sites such as iStation and Lexia.) This includes knowing not to disclose personal information such as last name, location, and passwords. Students practice giving positive feedback on other student posts. Digital citizenship is described with nine categories, however PreK-2 will focus on 4 of these: Digital Literacy (the ability to use new technology quickly and appropriately), Digital Etiquette (appropriate conduct), Digital Rights and Responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online), and Digital Health and Wellness (caring for your physical and psychological well-being online).		
1.IC.H.01	Compare how people live and work before and after the implementation or adoption of new computing technology. *As computers become interconnected in each aspect of society, more powerful, and students become more reliant on them, students should be able to identify a list of technologies the school and others have improved in their daily lives. (e.g., ordering devices by voice, financial institutions, household devices management, robotics, cars that drive themselves, and Social Media sharing applications.)*	History	7. Communicating about computing
1.IC.SLE.01	Practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software. Keep login information private, and log off of devices appropriately. *People use computing technology in ways that can help or hurt themselves or others. Harmful behaviors, such as sharing private information such as last name, location, and school, as well as leaving public devices logged in or sharing login information should be recognized and avoided. Students should understand they should never post as another person (blogs, SeeSaw, etc.) The concept of copyright and using photos and text with permission should be recognized and practiced with guidance.*	Safety, Law, & Ethics	2. Collaborating around computing

1.IC.CP.01	Compare and contrast examples of how computing technology has changed and improved the way people live, work, and interact. *Within the inevitable interwoven fabric of society's reliance and innovative machines, students will required to have basic assumable skills when entering the workforce. Students should be able to identify what digital computing devices and languages are necessary to create a modernized mode of everyday activities in the technological age. An example would be for students to list how a bus driver can use GPS, safety features, and indicators to provide safe travel to school.*	Community Partnerships	7. Communicating about computing

Second Grade

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
2.CS.D.01	Select and use a computing device to perform a variety of tasks for an intended outcome. *People use computing devices to perform a variety of tasks accurately and quickly. Students should be able to select the appropriate app/program to use for tasks they are required to complete, then log off or power down. For example, if students are asked to draw a picture, they should be able to open and use a drawing app/program to complete this task, or if they are asked to create a presentation, they should be able to open and use presentation software. In addition, with teacher guidance, students should compare and discuss preferences for software with the same primary functionality. Students could compare different web browsers or word processing, presentation, or drawing programs.*	Devices	7. Communicating about computing
2.CS.HS.01	Model the use of components of a computing system, its basic functions, peripherals, and storage features.(e.g. using the hard drive, memory/storage, printers, scanners, wireless and cabled connections, and cloud storage). *A computing system is composed of hardware and software. Hardware consists of physical components. Software provides a computer a set of instructions to follow. Students should be able to identify and use the function of software and hardware such as memory/storage, printers, flash drive, cloud storage, etc.*	Hardware and Software	7. Communicating about computing
2.CS.HS.02	Self-select and use appropriate software/apps for an intended outcome. (e.g., programs, browsers, websites, and applications). *Computer software and apps are programmed and installed on hard drives on various devices utilized by every end user. Software provides code for the programs to compute properly for the created operation. Software apps and programs interact with one another to provide an intended outcome or output. Students should be able to select an application or program required for a desired activity. This could include,*	Hardware and Software	1. Fostering an inclusive computing culture

	but not limited to, district purchased client-based reading or math program software, apps for a specific learning methods Reading Eggs, iMovie, Google Apps, Seesaw, or accessing a browser to navigate web based programs.		
2.CS.IO.01	Understand and use varying input/output skills. • Input (keyboarding, mouse, touchscreen, voice, voice typing, camera, robotics, interactive board) • Output (monitor, screen, printer, 3D printer, robotics, audio) *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice typing, touchpad, touchscreen, mouse, and keyboarding. (Keyboarding- use fingers on home row and the spacebar with the thumb, shift key for capital letters, understand that clicking the mouse or tapping the location on the screen makes an insertion point in a document and how to use the mouse to highlight (double-click) a word.*	Input and Output	7. Communicating about computing
2.CS.T.01	Using accurate terminology, identify and resolve simple hardware and software problems and strategies for solving these problems. *Problems with computing systems have different causes. Students at this level will start to understand those causes, and should be able to communicate a problem with accurate terminology, and be able to find solutions to that problem (e.g., when an app or program is not working as expected, a device will not turn on, the sound does not work, etc.). Ideally, students would be able to use simple troubleshooting strategies, including turning a device off and on to reboot it, closing and reopening an app, turning on speakers, or plugging in headphones. These are, however, not specified in the standard, because these problems may not occur.*	Troubleshooting	6. Testing and refining computational artifacts 7. Communicating about computing

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
2.NI.NCO.01	Use computing devices to share information and communicate with others using a network.	Network Communication & Organization	7. Communicating about computing

	Networking and interconnectivity of computing devices are essential in today's society. Through wi-fi, bluetooth, or hard line ethernet connections, the ability of information to be shared with an organized, secure and reliable system, is an integrated range of platforms which uses various software and hardware. Students should be able to understand and apply the process of sending information to the program or device (e.g., the teacher laptop is being connected to the LCD projector, or if the wi-fi or connection is active via Airplay, screenshare, airdrop, bluetooth speaker or headphones, Google Classroom uploads).		
2.NI.C.01	Demonstrate use of strong authentication methods to access and protect devices and data. Understand the effects of retaining password privacy. *Learning to protect one's device or information from unwanted use by others is an essential first step in learning about cybersecurity. Students are required to use strong passwords. They should appropriately use and protect the passwords they are required to use.*	Cybersecurity	7. Communicating about computing

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
2.DA.S.01	Manipulate existing files while use appropriate file-naming conventions. With guidance, develop and modify an organizational structure by creating, copying, moving, and deleting files and folders. *All information stored and processed by a computing device is referred to as data. Data can be images, text documents, audio files, software programs or apps, video files, etc. As students use software to complete tasks on a computing device, they will be manipulating data. Students will organize files or folders and use naming techniques (e.g., sorting content area activities, grouping photos by project, moving files or photos to the trash).*	Storage	4. Developing and using abstractions
2.DA.C.01	With guidance, collect and present the same data in various visual formats. *The collection and use of data about the world around them is a routine part of life and influences how people live. Students could collect data*	Collection	4. Developing and using abstractions 7. Communicating about computing

	on the weather, such as sunny days versus rainy days, the temperature at the beginning of the school day and end of the school day, or the inches of rain over the course of a storm. Students could count the number of pieces of each color of candy in a bag of candy, such as Skittles or M&Ms. Students could create surveys of things that interest them, such as favorite foods, pets, or TV shows, and collect answers to their surveys from their peers and others. The data collected could then be organized into two or more visualizations, such as a bar graph, pie chart, or pictograph.		
2.DA.CVT.01	Collect data over time and organize it on a chart or graph in order to make a prediction. *Data can be used to make inferences or predictions about the world. Students could analyze a graph or pie chart of the colors in a bag of candy or the averages for colors in multiple bags of candy, identify the patterns for which colors are most and least represented, and then make a prediction as to which colors will have most and least in a new bag of candy. Students collect data over time, then create and analyze charts or graphs in spreadsheet applications, web based programs, or visually in digital drawings to portray data collected. They could create and analyze graphs of temperatures taken at the beginning of the school day and end of the school day, identify the patterns of when temperatures rise and fall, and predict if they think the temperature will rise or fall at a particular time of the day, based on the pattern observed. The focus is on organizing data and making predictions based on data.*	Visualization & Transformation	7. Communicating about computing
2.DA.IM.01	Use patterns in data to make inferences or predictions based on data collected from users or simulations. *Data can be represented in models to portray results and to assist in identifying patterns in the world around us. This includes students collecting their own data or experiencing digital simulations. The intent is to make predictions based on the data collected from participants or from simulations.*	Inference and Models	4. Developing and using abstractions

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
2.AP.A.01	Both independently and collaboratively construct and follow algorithms that include sequencing and simple loops to accomplish a task verbally, kinesthetically, with robot devices, or a programing language. *Algorithmic thinking is the ability to define clear steps to solve a problem. Composition is the combination of smaller tasks into more complex tasks. With guidance, students should be able to create and follow algorithms for making simple foods, brushing their teeth, getting ready for school, participating in clean-up time or programming a robotic device to follow a preset path. Students should understand that loops repeat the steps of a process.*	Algorithms	4. Developing and using abstractions
2.AP.V.01	Use and model the way a computer program stores, accesses, and manipulates data that is represented as a variable. *Information in the real world can be represented in computer programs. Students could use thumbs up/down as representations of yes/no, use arrows when writing algorithms to represent direction, use emojis that represent emotion, or use common icons and symbols to perform an action (play is a triangle, save button, share button, etc.).*	Variables	4. Developing and using abstractions
2.AP.C.01	Independently and collaboratively create programs to accomplish tasks using a programming language such as block based programming using a robot device, or unplugged activity that includes simple loops, sequencing, and repetition. *Programming is used as a tool to create products that reflect a wide range of interests. Control structures specify the order in which instructions are executed within a program. Use block based programming, which is found in most robots used in elementary schools, or online resources to learn coding skills.*	Control	5. Creating computational artifacts
2.AP.M.01	Independently decompose (break down) a larger problem into smaller subproblems and steps needed to solve those problems. *Decomposition is the act of breaking down tasks into simpler tasks. Students could break down the steps needed to make a peanut butter*	Modularity	3. Recognizing and defining computational problems

	and jelly sandwich, to brush their teeth, to draw a shape, to move a character across the screen, or to solve a level of a coding app. When coding, including the setting, designing a character, and choosing the actions.		
2.AP.PD.01	Independently 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). *Creating a plan for what a program will do clarifies the steps that will be needed to create a program and can be used to check if a program is correct. Students could create a planning document, such as a story map, a storyboard, or a sequential graphic organizer, to illustrate what their program will do. Students at this stage should be able to complete the planning process by themselves.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
2.AP.PD.02	Give credit to ideas, creation (such as code, music, or pictures) and solutions of others while writing and developing programs. *Using computers comes with a level of responsibility. Students should credit artifacts that were created by others, such as pictures, music, and code. Credit could be given orally, if presenting their work to the class, or in writing or orally, if sharing work on a class blog or website. Proper attribution at this stage does not require a formal citation, such as in a bibliography or works cited document.*	Program Development	7. Communicating about computing
2.AP.PD.03	Independently and collaboratively construct, execute, analyze and debug (fix) an algorithm using a programming language and/or unplugged activity that includes sequencing and simple loops. *Algorithms or programs may not always work correctly. Students should be able to independently use various strategies, such as changing the sequence of the steps, following the algorithm in a step-by-step manner, or trial and error to fix problems in algorithms and programs.*	Program Development	6. Testing and refining computational artifacts
2.AP.PD.04	Use correct terminology (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.	Program Development	7. Communicating about computing

	At this stage, students should be able to use correct terminology to discuss or write about the goals and expected outcomes of the programs they create and the choices that they made when creating programs. This could be done using coding journals, discussions with a teacher, class presentations, or blogs.		

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
2.IC.C.01	Recognize and describe how different technologies used daily in work and at home are used to solve problems or make work and life easier. *Computing technology has changed the way people live and work. In the past, if students wanted to read about a topic, they needed access to a library to find a book about it. Today, students will be able to view and read information on the Internet about a topic or they can download e-books about it directly to a device. Such information may be available in more than one language and could be read to a student, allowing for great accessibility. In their personal lives, they should be able to utilize those same technologies to complete life tasks (e.g., ordering food, video-editing, game play, drones, sending emails to family and friends for social interactions, and possibly checking the weather for the next day).*	Culture	7. Communicating about computing
2.IC.SI.01	Aid in developing an appropriate code of conduct, explain and practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior (Digital Citizenship - review Digital Literacy and Digital Etiquette, but focus on Rights and Responsibilities and Digital Health and Wellness). *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Students could share their work on blogs or in other collaborative spaces online, taking care to avoid sharing information that is inappropriate or that could personally identify them to others. Students could provide feedback to others on their work in a kind and respectful manner and could tell an adult if others are sharing things they should*	Social Interactions	2. Collaborating around computing

	not share or are treating others in an unkind or disrespectful manner on online collaborative spaces. Digital citizenship is described with nine categories, however PreK-2 will focus on 4 of these: Digital Literacy (the ability to use new technology quickly and appropriately), Digital Etiquette (appropriate conduct), Digital Rights and Responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online), and Digital Health and Wellness (caring for your physical and psychological well-being online).		
2.IC.H.01	Recognize how technologies have changed the world, and explore how the needs of society have impacted the changes in technology. *As computers become interconnected in each aspect of society, more powerful, and students become more reliant on them, students should be able to explain or utilize a list of technologies the school and others have improved in their daily lives making connections to real-world problems and solutions. (e.g., ordering devices by voice, financial institutions, medical fields, household devices management, robotics, cars that drive themselves, and Social Media sharing applications.)*	History	7. Communicating about computing
2.IC.SLE.01	Practice responsible digital citizenship in all technology use. Understand digital data has intellectual property rights (belongs to others) and it cannot be claimed as your own. *People use computing technology in ways that can help or hurt themselves or others. Harmful behaviors, such as sharing private information or sharing login information should be recognized and avoided. Students should understand they should never post as another person (blogs, SeeSaw, etc.) Students should be aware of the concept of copyright and using photos and text with permission. This could include images online, or asking a friend if it is OK to post their picture before sharing it digitally.*	Safety, Law, & Ethics	2. Collaborating around computing
2.IC.CP.01	Investigate how computer science has impacted your daily life and the jobs in your community and the world around you. *Within the inevitable interwoven fabric of society's reliance and innovative machines, students will required to have basic assumable skills when entering the workforce. Students should be able to explain*	Community Partnerships	7. Communicating about computing

	how digital computing devices and languages are necessary to create a modernized mode of everyday activities in the technological age. An example would be for students to create examples and give possible improvements of how a bus driver can use GPS, safety features, and indicators to provide safe travel to school.		

Third Grade

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
3.CS.D.01	Identify how computing devices can be connected to other devices to extend their capabilities. *Computing devices often depend on other devices or components. For example, a robot depends on a physically attached light sensor to detect changes in brightness, whereas the light sensor depends on the robot to power. Keyboard input or a mouse click could cause an action to happen or information to be displayed on a screen; this could only happen because the computer has a processor to evaluate what is happening externally and produce corresponding responses. At this stage, students should be able to identify basic connections of a minimum of two components (such as a tablet and charger cable functioning together to charge the device or connect to the computer for sharing data) while learning correct terminology for these devices and components.*	Devices	7. Communicating about computing
3.CS.HS.01	Model how information flows through hardware and software to accomplish tasks. *In order for a person to accomplish tasks with a computer, both hardware and software are needed. At this stage, a model should only include basic elements of a computer system, such as input, output, processor, sensors, and storage. Students could draw a model on paper or in a drawing program.*	Hardware and Software	4. Developing and using abstractions
3.CS.IO.01	Demonstrate proper use of grade level appropriate input devices and produce digital artifacts with a controlled audience. *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice typing, touchpad, touchscreen, mouse, keyboarding (type letters and words at a rate of 5-10 WPM while looking, identify home row, modifier, punctuation, function keys), audio devices, camera. Digital artifacts could be published but within a controlled setting like a closed class blog or website. Examples of digital artifacts could include a slideshow,*	Input and Output	7. Communicating about computing

	video, prints, 3D prints, audio, programs (robotics), web-based product (controlled audience).		
3.CS.T.01	Identify, using accurate terminology, simple hardware and software problems and strategies for solving these problems. *Although computing systems may vary, common troubleshooting strategies can be used on all of them. Students should be able to identify solutions to problems such as the device not responding, no power, no network, app crashing, no sound, or password entry not working. Should an error occur, the goal would be that students would identify various strategies, such as rebooting the device, checking for power, checking network availability, closing and reopening applications, making sure the volume is turned up and the headphones are plugged in, and making sure the caps lock key is not on, to solve these problems, when possible. It also becomes crucial for students to start using accurate terminology in describing and discussing their problem with a peer or adult.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
3.NI.NCO.01	Model how a device on a network sends and receives information. *Information is sent and received over physical or wireless paths. It is broken down into smaller pieces called packets, which are sent independently and reassembled at the destination. Students should demonstrate their understanding of this flow of information, e.g. drawing a model of the way packets are transmitted, programming an animation to show how packets are transmitted, or demonstrating through an unplugged activity which has them act it out in some way.*	Network Communication & Organization	4. Developing and using abstractions
3.NI.C.01	Identify problems that relate to inappropriate use of computing devices and networks. *Just as we protect our personal property offline, we also need to protect our devices and the information stored on them. Information can be protected using various security measures. These measures can be physical and/or digital. Students could discuss or use a journaling or*	Cybersecurity	3. Recognizing and Defining Computational Problems

	blogging activity to explain, orally or in writing, about topics that relate to personal cybersecurity issues. Discussion could be based on topics that are applicable to students, such as backing up data to guard against loss, how to create strong passwords and the importance of not sharing passwords. or why we should install and keep anti-virus software updated to protect data and systems..		

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
3.DA.S.01	Compare and contrast the formats and storage requirements for different types of information (e.g., music, video, images, and text). *Different Software tools used to access data may store the data differently. The type of data being stored and the level of detail represented by that data affect the storage requirements (file size, availability, and available memory). Music, images, video, and text require different amounts of storage. Video will often require more storage than music or images alone because video combines both.*	Storage	7. Communicating about computing
3.DA.C.01	Gather relevant and reliable data to solve a problem or answer a question. *People select digital tools for the collection of data based on what is being observed and how the data will be used (e.g., a thermometer is used to measure temperature and GPS sensor is used to track locations). There exists a wide array of digital data collection tools, and only some are appropriate for certain types of data. Tools are chosen based upon the type of measurement they use as well the type of data people wish to observe.*	Collection	5. Creating Computational Artifacts
3.DA.CVT.01	Create a simple data visualization based on data collected by or provided to student. *Raw data has little meaning on its own. Data is often sorted or grouped to provide additional clarity. Organizing data can make interpreting and communicating it to others easier. Data points can be clustered by a number of commonalities. The same data could be manipulated in different ways to emphasize particular aspects or parts of the data set*	Visualization & Transformation	7. Communicating about computing

	(e.g., graphs, charts and infographics). For example, a data set of sports teams could be sorted by wins, points scored, or points allowed, and a data set of weather information could be sorted by high temperatures, low temperatures, or precipitation.		
3.DA.IM.01	Utilize data to make predictions and discuss whether there is adequate data to make reliable predictions. *The accuracy of data analysis is related to how realistically data is represented. Inferences or predictions based on data are less likely to be accurate if the data is not sufficient or if the data is incorrect in some way. Students should be able to refer to data when communicating an idea. For example, in order to explore the relationship between speed, time, and distance, students could operate a robot at uniform speed, and at increasing time intervals to predict how far the robot travels at that speed. In order to make an accurate prediction, one or two attempts of differing times would not be enough. The robot may also collect temperature data from a sensor, but that data would not be relevant for the task. Students must also make accurate measurements of the distance the robot travels in order to develop a valid prediction. Students could record the temperature at noon each day as a basis to show that temperatures are higher in certain months of the year. If temperatures are not recorded on non-school days or are recorded incorrectly or at different times of the day, the data would be incomplete and the ideas being communicated could be inaccurate. Students may also record the day of the week on which the data was collected, but this would have no relevance to whether temperatures are higher or lower. In order to have sufficient and accurate data on which to communicate the idea, students might want to use data provided by a governmental weather agency.*	Inference and Models	7. Communicating about computing

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
3.AP.A.01	Compare multiple algorithms for the same task. *Different algorithms can achieve the same result; however, sometimes one algorithm might be more suited for a particular situation. Students should be able to look at different ways to solve the problem or complete*	Algorithms	3. Recognizing and Defining Computational Problems

	the same task and recognize the differences between the solutions. For example, students could create multiple algorithms that describe how to get ready for school or other tasks like baking cookies.		6. Testing and Refining Computational Artifacts
3.AP.V.01	Utilize simple programs that use variables to store and modify grade level appropriate data. *Variables are used to store and modify data. At this level, understanding how to use variables is sufficient. For example, students may use mathematical operations to add to the score of a game or subtract from the number of lives available in a game. The use of a variable is a countdown timer is another example.*	Variables	5. Creating computational artifacts
3.AP.C.01	Create simple programs using a programming language that utilize sequencing, repetition, conditionals, and variables to solve a problem or express ideas independently. *Control structures specify the order (sequence) in which instructions are executed within a program and can be combined to support the creation of more complex programs. Conditionals allow for the execution of a portion of code in a program when a certain condition is true. For example, students could write a math game that asks for multiplication fact questions and then uses a conditional to check whether or not the answer that was entered is correct. Loops allow for the repetition of a sequence of code multiple times. For example, in a program that produces an animation about a famous historical character, students could use a loop to have the character walk across the screen as they introduce themselves.*	Control	5. Creating computational artifacts
3.AP.M.01	Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. *Students should be able to take a general solution to a problem and break down steps that are too generic. For example, baking a cake could be described in various levels of detail. Many steps, like adding ingredients to a bowl, can be broken down into multiple steps instead of just adding all ingredients at once.*	Modularity	3. Recognizing and defining computational problems

3.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. *Programs can be broken down into smaller parts, which can be incorporated into new or existing programs. For example, students could modify prewritten code from a single-player game to create a two-player game with slightly different rules, remix and add another scene to an animated story, use code to make a ball bounce from another program in a new basketball game, or modify an image created by another student.*	Modularity	3. Recognizing and defining computational problems 5. Creating computational artifacts
3.AP.PD.01	Create a plan using an iterative process to plan the development of a program while solving simple problems (e.g., storyboard, flowchart, pseudo-code, story map). *Students should document the plan development as, for example, a storyboard, flowchart, pseudocode, or story map. Students put commands in order (ties into literacy and expository text) (e.g. using block code to drag commands into the correct order to complete the programming task.*	Program Development	1. Fostering an Inclusive Computing Culture 5. Creating computational artifacts
3.AP.PD.02	Use proper citations and document when ideas are borrowed and changed for their own use (e.g., using pictures created by others, using music created by others, remixing programming projects). *Students should identify instances of remixing, when ideas are borrowed and iterated upon, and credit the original creator. Students should also consider common licenses that place limitations or restrictions on the use of computational artifacts, such as images and music downloaded from the Internet. At this stage, attribution should be written in the format required by the teacher. (e.g. as students begin using resources created by others a first step in writing citations is collecting the website link from where you found your artifact.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
3.AP.PD.03	Analyze and debug (identify/fix errors) a program that includes sequencing, repetition and variables in a programming language.	Program Development	1. Fostering an Inclusive Computing Culture

	As students develop programs they should continuously test those programs to see that they do what was expected and fix (debug), any errors. Students should also be able to assist others in debugging their programs.		2. Collaborating Around Computing 6. Testing and Refining Computational Artifacts
3.AP.PD.04	Communicate and explain your program development using comments, presentations and demonstrations. *People communicate about their code to help others understand and use their programs. Another purpose of communicating one's design choices is to show an understanding of one's work. These explanations could be in-line code comments or as part of a summative presentation, such as a code walk-through or coding journal.*	Program Development	2. Collaborating Around Computing 7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
3.IC.C.01	Identify possible problems and how computing devices have built in features for increasing accessibility to all users. *Anticipating the needs and wants of diverse end users requires students to purposefully consider potential perspectives of users with different backgrounds, ability levels, points of view, and disabilities. For example, students may consider using both speech and text to convey information in a game. They may also wish to vary the types of programs they create, knowing that not everyone shares their own tastes. When creating something for others, students give options (e.g. speech to text or type, differentiate tasks, adjusting hardware needed/give options because others might not all have the same tools.*	Culture	4. Developing and Using Abstractions 5. Creating Computational Artifacts 6. Testing and Refining Computational Artifacts
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. (Digital Citizenship - review of all nine components, but focused on Digital Communication and Digital Etiquette.)	Social Interactions	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing

	The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Digital citizenship is described with nine categories: digital access (awareness of materials gained online and being mindful of who does/does not have access), digital commerce (awareness of illegal and legal exchanges online e.g. illegal downloading), digital communication (communicating and collaborating properly online), digital literacy (the ability to use new technology quickly and appropriately), digital etiquette (appropriate conduct), digital law (ethical use of technology e.g. hacking information, downloading illegally, plagiarizing, creating viruses, sending spam, or stealing someone's identify), digital rights and responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online) digital health and wellness (caring for your physical and psychological wellbeing online), digital security (proactive about protecting your devices and identity online e.g., data backup, use of a surge protector, virus protection).		
3.IC.S.02	Identify how computational products may be, or have been, improved to incorporate diverse perspectives. *Computing provides the possibility for collaboration and sharing of ideas and allows the benefit of diverse perspectives. For example, students could seek feedback from other groups in their class or students at another grade level. (e.g., Students could begin by evaluating lesson materials saying, "This assignment would be better if..." You color coded, made this assignment in a table, made it accessible on my mom's phone.")*	Social Interactions	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing
3.IC.H.01	Identify computing technologies that have changed the world, and express how those technologies influence, and are influenced by, society. *Students, with guidance from their teacher, should discuss topics that relate to the history of technology and the changes in the world due to technology. Topics could be based on current news content, such as robotics, wireless Internet, mobile computing devices, GPS systems, wearable computing, or how social media has influenced social and*	History	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing

	political changes. (e.g. Google Glasses, Robotics, 3D printers, smart phones, Chromebooks, Precision Ag (lettucebot, GPS tractors, boom cameras) drones, Smart assistants, Students and teacher create a collaborative list of current technology and discuss the impacts those devices have on our lives. Also, discuss why devices are popular/unpopular and why, which is how that device is viewed by society.		
3.IC.SLE.01	Identify types of digital data that may have intellectual property rights that prevent copying or require attribution. *Students should consider the licenses on computational artifacts that they wish to use. For example, the license on a downloaded image on audio file may have restrictions that prohibit modification, require attribution, or prohibit use entirely. Students should have a basic knowledge of items that are restricted. (e.g., online books, music, free music archive, images, creative commons).*	Safety, Law, & Ethics	5. Creating Computational Artifacts
3.IC.CP.01	Design a visual product depicting the connections between computer science and other fields. *Explaining the reason why of any computer task will lead students to understand how other professionals within their community might use similar tasks in their occupations. Making correlations and a purpose for tasks makes CS relevant to their lives as they age. Students make a direct correlation to a local business.*	Community Partnerships	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing 7. Communicating about computing

Fourth Grade

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
4.CS.D.01	Identify and explain how computing devices can be connected to other devices to extend their capabilities. *Computing devices often depend on other devices or components. For example, a robot depends on a physically attached light sensor to detect changes in brightness, whereas the light sensor depends on the robot to power. Keyboard input or a mouse click could cause an action to happen or information to be displayed on a screen; this could only happen because the computer has a processor to evaluate what is happening externally and produce corresponding responses. Students should be able to identify connections of a minimum of three components (such as a computer charger connected to a computer for power and then connecting to wifi through an access point within the vicinity) and explain how devices and components interact using correct terminology.*	Devices	7. Communicating about computing
4.CS.HS.01	Explain how information is translated, transmitted, and processed between hardware and software in order to accomplish tasks. *In order for a person to accomplish tasks with a computer, both hardware and software are needed. At this stage, a model should only include basic elements of a computer system, such as input, output, processor, sensors, and storage. Students could draw a model in a drawing program, program an animation to demonstrate it, or demonstrate it by acting this out in some way.*	Hardware and Software	4. Developing and using abstractions
4.CS.IO.01	Demonstrate proper use of grade level appropriate input devices and produce digital artifacts with a controlled audience. *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice typing, touchpad, touchscreen, mouse, keyboarding (type letters and words at a rate of 10-15 WPM while increasing the proportion of time looking away from the keyboard, consistent use of home row, modifier, punctuation, function keys), audio devices, camera. Digital artifacts could be*	Input and Output	7. Communicating about computing

	published but within a controlled setting like a closed class blog or website. Examples of digital artifacts could include a slideshow, video, prints, 3D prints, audio, programs (robotics), web-based product (controlled audience).		
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. *Although computing systems may vary, common troubleshooting strategies can be used on all of them. Students should be able to identify solutions to problems such as the device not responding, no power, no network, app crashing, no sound, or password entry not working. Should an errors occur, the goal would be that students would not only identify but also use various strategies, such as rebooting the device, checking for power, checking network availability, closing and reopening applications, making sure the volume is turned up and the headphones are plugged in, and making sure the caps lock key is not on, to solve these problems, when possible. Students would continue using and build on accurate terminology in describing and discussing their problem with a peer or adult.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
4.NI.NCO.01	Explain how information is sent and received across physical or wireless paths. *Information is sent and received over physical or wireless paths. It is broken down into smaller pieces called packets, which are sent independently and reassembled at the destination. Students should demonstrate their understanding of this flow of information, e.g., drawing a model of the way packets are transmitted, programming an animation to show how packets are transmitted, or demonstrating through an unplugged activity which has them act it out in some way.*	Network Communication & Organization	4. Developing and using abstractions
4.NI.C.01	Identify and explain issues related to responsible use of technology and information, and describe personal consequences of inappropriate use.	Cybersecurity	3. Recognizing and Defining

	Just as we protect our personal property offline, we also need to protect our devices and the information stored on them. Information can be protected using various security measures. These measures can be physical and/or digital. Students could discuss or use a journaling or blogging activity to explain, orally or in writing, about topics that relate to personal cybersecurity issues. Discussion topics could be based on current events related to cybersecurity or topics that are applicable to students, such as backing up data to guard against loss, how to create strong passwords and the importance of not sharing passwords. or why we should install and keep anti-virus software updated to protect data and systems.		Computational Problems

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
4.DA.S.01	Classify different storage locations (physical, shared, or cloud) based on the type of file, storage requirements, and sharing requirements. *Different Software tools used to access data may store the data differently. The type of data being stored and the level of detail represented by that data affect the storage requirements (file size, availability, and available memory). Music, images, video, and text require different amounts of storage. Video will often require more storage than music or images alone because video combines both.*	Storage	7. Communicating about computing
4.DA.C.01	Gather and manipulate relevant and reliable data using the appropriate digital tool. *People select digital tools for the collection of data based on what is being observed and how the data will be used (e.g., thermometer is used to measure temperature and GPS sensor is used to track locations). There is a wide array of digital data collection tools, only some are appropriate for certain types of data. Tools are chosen based upon the type of measurement they use as well the type of data people wish to observe.*	Collection	5. Creating Computational Artifacts
4.DA.CVT.01	Organize and present collected data visually to highlight comparisons.	Visualization & Transformation	7. Communicating about computing

	Raw data has little meaning on its own. Data is often sorted or grouped to provide additional clarity. Organizing data can make interpreting and communicating it to others easier. Data points can be clustered by a number of commonalities. The same data could be manipulated in different ways to emphasize particular aspects or parts of the data set (e.g., graphs, charts and infographics). For example, a data set of sports teams could be sorted by wins, points scored, or points allowed, and a data set of weather information could be sorted by high temperatures, low temperatures, or precipitation.		
4.DA.IM.01	Determine how the accuracy of conclusions are influenced by the amount and relevance of the data collected. *The accuracy of data analysis is related to how realistically data is represented. Inferences or predictions based on data are less likely to be accurate if the data is not sufficient or if the data is incorrect in some way. Students should be able to refer to data when communicating an idea. For example, in order to explore the relationship between speed, time, and distance, students could operate a robot at uniform speed, and at increasing time intervals to predict how far the robot travels at that speed. In order to make an accurate prediction, one or two attempts of differing times would not be enough. The robot may also collect temperature data from a sensor, but that data would not be relevant for the task. Students must also make accurate measurements of the distance the robot travels in order to develop a valid prediction. Students could record the temperature at noon each day as a basis to show that temperatures are higher in certain months of the year. If temperatures are not recorded on non-school days or are recorded incorrectly or at different times of the day, the data would be incomplete and the ideas being communicated could be inaccurate. Students may also record the day of the week on which the data was collected, but this would have no relevance to whether temperatures are higher or lower. In order to have sufficient and accurate data on which to communicate the idea, students might want to use data provided by a governmental weather agency.*	Inference and Models	7. Communicating about computing

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
4.AP.A.01	Analyze and refine multiple algorithms for the same task. *Different algorithms can achieve the same result; however, sometimes one algorithm might be more suited for a particular situation. Students should be able to look at different ways to solve a problem or complete a task and decide which would be the best solution. For example, students could write different algorithms to draw a regular polygon and determine which algorithm would be the easiest to modify or repurpose to draw a different polygon.*	Algorithms	3. Recognizing and Defining Computational Problems 6. Testing and Refining Computational Artifacts
4.AP.V.01	Utilize, create, and modify programs that use variables, with grade level appropriate data. *Variables are used to store and modify data. At this level, understanding how to use variables in a variety of ways is sufficient. For example, students may use mathematical operations to add to the score of a game or subtract from the number of lives available in a game. The use of a variable in a countdown timer is another example.*	Variables	5. Creating computational artifacts
4.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. *Control structures specify the order (sequence) in which instructions are executed within a program and can be combined to support the creation of more complex programs. Conditionals allow for the execution of a portion of code in a program when a certain condition is true. For example, students could write a math game that asks for multiplication fact questions and then uses a conditional to check whether or not the answer that was entered is correct. Loops allow for the repetition of a sequence of code multiple times. For example, in a program that produces an animation about a famous historical character, students could use a loop to have the character walk across the screen as they introduce themselves. Students should be able to complete these tasks collaboratively with other students.*	Control	5. Creating computational artifacts

4.AP.M.01	Decompose (break down) large problems into smaller, manageable subproblems. Then form algorithms to solve each subproblem. *Decomposition is the act of breaking down tasks into simpler tasks. For example, students could create an animation by separating a story into different scenes. For each scene, they would select a background, place characters, and describe actions.*	Modularity	3. Recognizing and defining computational problems
4.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. *Programs can be broken down into smaller parts, which can be incorporated into new or existing programs. For example, students could modify prewritten code from a single-player game to create a two-player game with slightly different rules, remix and add another scene to an animated story, use code to make a ball bounce from another program in a new basketball game, or modify an image created by another student.*	Modularity	3. Recognizing and defining computational problems 5. Creating computational artifacts
4.AP.PD.01	Create a plan using an iterative process to plan the development of a program that includes user preferences while solving simple problems. *Planning is an important part of the iterative process of program development. Students outline features, time and resource constraints, and user expectations. Students should document the plan as, for example, a storyboard, flowchart, pseudocode, or story map.*	Program Development	1. Fostering an Inclusive Computing Culture 5. Creating computational artifacts
4.AP.PD.02	Use proper citations and document when ideas are borrowed and changed for their own use (e.g., using pictures created by others, using music created by others, remixing programming projects). *Students should identify instances of remixing, when ideas are borrowed and iterated upon, and credit the original creator. Students should also consider common licenses that place limitations or restrictions on the use of computational artifacts, such as images and music downloaded from the Internet. At this stage, attribution should be written in the format required by the teacher and should always be included on any programs shared online. Students work through how to create citations for various*	Program Development	5. Creating computational artifacts 7. Communicating about computing

	borrowed resources. both writing citations manually then introducing websites that assist in citation creation.		
4.AP.PD.03	Analyze, debug (identify/fix errors), and create a program that includes sequencing, repetition and variables in a programming language. *As students develop programs they should continuously test those programs to see that they do what was expected and fix (debug), any errors. Students should also be able to successfully find simple errors in programs created by others.*	Program Development	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing 6. Testing and Refining Computational Artifacts
4.AP.PD.04	Communicate and explain your program development using comments, presentations and demonstrations. *People communicate about their code to help others understand and use their programs. Another purpose of communicating one's design choices is to show an understanding of one's work. These explanations could manifest themselves as in-line code comments for collaborators and assessors, or as part of a summative presentation, such as a code walk-through or coding journal.*	Program Development	2. Collaborating Around Computing 7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
4.IC.C.01	Brainstorm problems and ways to improve computing devices to increase accessibility to all users. *Anticipating the needs and wants of diverse end users requires students to purposefully consider potential perspectives of users with different backgrounds, ability levels, points of view, and disabilities. For example, students may consider using both speech and text to convey information in a game. They may also wish to vary the types of programs they create, knowing that not everyone shares their own tastes. When creating something for others, students give options (e.g. speech to text or type, differentiate tasks, adjusting hardware needed/give options because others might not all have the same tools.*	Culture	4. Developing and Using Abstractions 5. Creating Computational Artifacts 6. Testing and Refining Computational Artifacts

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. (Digital Citizenship - review of all nine components, but focused on Digital Access). *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Digital citizenship is described with nine categories: digital access (awareness of materials gained online and being mindful of who does/does not have access), digital commerce (awareness of illegal and legal exchanges online e.g. illegal downloading), digital communication (communicating and collaborating properly online), digital literacy (the ability to use new technology quickly and appropriately), digital etiquette (appropriate conduct), digital law (ethical use of technology e.g. hacking information, downloading illegally, plagiarizing, creating viruses, sending spam, or stealing someone's identify), digital rights and responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online) digital health and wellness (caring for your physical and psychological wellbeing online), digital security (proactive about protecting your devices and identity online e.g. data backup, use of a surge protector, virus protection).*	Social Interactions	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing
4.IC.SI.02	As a team, consider each other's' perspectives on improving a computational product. *Computing provides the possibility for collaboration and sharing of ideas and allows the benefit of diverse perspectives. For example, students could seek feedback from other groups in their class or students at another grade level.*	Social Interactions	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing
4.IC.H.01	Identify and give examples of computing technologies that have changed the world, and express how those technologies influence, and are influenced by, society. *Students, with guidance from their teacher, should discuss topics that relate to the history of technology and the changes in the world due to*	History	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing

	technology. Topics could be based on current news content, such as robotics, wireless Internet, mobile computing devices, GPS systems, wearable computing, or how social media has influenced social and political changes. (e.g. a collaborative table with headings - Tech, Reason for the Tech, and Society Changes with this Tech.) This could be a collaborative activity where everyone adds their own ideas.		
4.IC.SLE.01	Discuss the social impact of violating intellectual property rights. *Students should consider the licenses on computational artifacts that they wish to use. For example, the license on a downloaded image or audio file may have restrictions that prohibit modification, require attribution, or prohibit use entirely. Students should identify an artifact labeled not for reuse and explain why they should not use it and what might happen if they violated those restrictions.*	Safety, Law, & Ethics	5. Creating Computational Artifacts
4.IC.CP.01	Design a visual product depicting the connections between computer science and other fields. *Explaining the reason why of any computer task will lead students to understand how other professionals within their community might use similar tasks in their occupations. Making correlations and a purpose for tasks makes CS relevant to their lives as they age. Students make a direct correlation to a local business.*	Community Partnerships	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing 7. Communicating about computing

Fifth Grade

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
5.CS.D.01	Model and communicate how computing devices can be connected to other devices to extend their capabilities. *Students should have enough basic knowledge to identify examples of computing devices and components working together. To advance this knowledge, students would now communicate their understanding through a variety of means such as the creation of a slideshow, video, drawing, animation or other digital product depicting new examples of how computing devices can be connected to other devices to extend their capabilities. The intent of the completed student products would be to use them during instruction of the device standard with younger grade levels.*	Devices	7. Communicating about computing
5.CS.HS.01	Illustrate how information is translated into binary numbers between software and hardware. *Students should understand that everything on a computer can be reduced to1's and 0's (binary). That is, information they use and create as part of their programs, on the internet, and other devices are not stored on hardware in their apparent form, but as a series of binary codes. Students should be able to understand basic binary representation and how it can be used to store information. This can start with simple representation of on/off with lights (1/0 for current/no current) and gradually grow into how you can use a series of binary numbers to represent different kinds of information like text or numbers.*	Hardware and Software	4. Developing and using abstractions
5.CS.IO.01	Demonstrate proper use of grade level appropriate input devices and produce digital artifacts selective publication based on audience/purpose. *Input devices are used to input data for the creation of various digital products. Some input devices a person could use include voice typing, touchpad, touchscreen, mouse, keyboarding (type letters and words at a rate of 15-20 WPM with 85% accuracy while looking away from the keyboard, consistent use of home row, modifier, punctuation, function*	Input and Output	7. Communicating about computing

	keys), audio devices, camera. Digital artifacts would be published for the purpose of sharing with the appropriate audience based on the purpose of the artifact. Examples of digital artifacts could include a slideshow, video, prints, 3D prints, audio, programs (robotics), web-based product.		
5.CS.T.01	Using accurate terminology, identify simple hardware and software problems that may occur during everyday use. *Although computing systems may vary, common troubleshooting strategies can be used on all of them. Students should be able to identify solutions to problems such as the device not responding, no power, no network, app crashing, no sound, or password entry not working. Should an errors occur, the goal would be that students would not only identify but also use various strategies, such as rebooting the device, checking for power, checking network availability, closing and reopening applications, making sure the volume is turned up and the headphones are plugged in, and making sure the caps lock key is not on, to solve these problems, when possible. Students would continue using and build on accurate terminology in discussing their problem with a peer or adult.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
5.NI.NCO.01	Model how information is broken down into smaller pieces and transmitted through multiple devices over networks and the internet, and how these pieces are assembled at the destination. *Information is sent and received over physical or wireless paths. It is broken down into smaller pieces called packets, which are sent independently and reassembled at the destination. Students should demonstrate their understanding of this flow of information, e.g., drawing a model of the way packets are transmitted, programming an animation to show how packets are transmitted, or demonstrating through an unplugged activity which has them act it out in some way.*	Network Communication & Organization	4. Developing and using abstractions
5.NI.C.01	Discuss real-world cybersecurity problems and identify strategies for how personal information can be protected.	Cybersecurity	3. Recognizing and Defining

	Just as we protect our personal property offline, we also need to protect our devices and the information stored on them. Information can be protected using various security measures. These measures can be physical and/or digital. Students could discuss or use a journaling or blogging activity to explain, orally or in writing, about topics that relate to personal cybersecurity issues. Discussion topics could be based on current events related to cybersecurity or topics that are applicable to students, such as backing up data to guard against loss, how to create strong passwords and the importance of not sharing passwords, or why we should install and keep anti-virus software updated to protect data and systems..		Computational Problems

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
5.DA.S.01	Evaluate trade-offs of file types, storage requirements, and sharing requirements, including comparisons of availability and quality. *Different software tools used to access data may store the data differently. The type of data being stored and the level of detail represented by that data affect the storage requirements. Music, images, video, and text require different amounts of storage. Video will often require more storage than music or images alone because video combines both. For example, two pictures of the same object can require different amounts of storage based upon their resolution. Different software tools used to access and store data may add additional data about the data (metadata), which results in different storage requirements.*	Storage	7. Communicating about computing
5.DA.C.01	Select the appropriate tool to collect relevant and reliable data that solves a problem. *People select digital tools for the collection of data based on what is being observed and how the data will be used (e.g., thermometer is used to measure temperature and GPS sensor is used to track locations). There is a wide array of digital data collection tools, only some are appropriate for certain types of data. Tools are chosen based*	Collection	5. Creating Computational Artifacts

	upon the type of measurement they use as well the type of data people wish to observe.		
5.DA.CVT.01	Organize and present collected data to highlight comparisons and support a claim. *Raw data has little meaning on its own. Data is often sorted or grouped to provide additional clarity. Organizing data can make interpreting and communicating it to others easier. Data points can be clustered by a number of commonalities. The same data could be manipulated in different ways to emphasize particular aspects or parts of the data set. For example, a data set of sports teams could be sorted by wins, points scored, or points allowed, and a data set of weather information could be sorted by high temperatures, low temperatures, or precipitation.*	Visualization & Transformation	7. Communicating about computing
5.DA.IM.01	Use data to discover or propose cause and effect relationships, predict outcomes, or communicate an idea. *The accuracy of data analysis is related to how realistically data is represented. Inferences or predictions based on data are less likely to be accurate if the data is not sufficient or if the data is incorrect in some way. Students should be able to refer to data when communicating an idea. For example, in order to explore the relationship between speed, time, and distance, students could operate a robot at uniform speed, and at increasing time intervals to predict how far the robot travels at that speed. In order to make an accurate prediction, one or two attempts of differing times would not be enough. The robot may also collect temperature data from a sensor, but that data would not be relevant for the task. Students must also make accurate measurements of the distance the robot travels in order to develop a valid prediction. Students could record the temperature at noon each day as a basis to show that temperatures are higher in certain months of the year. If temperatures are not recorded on non-school days or are recorded incorrectly or at different times of the day, the data would be incomplete and the ideas being communicated could be inaccurate. Students may also record the day of the week on which the data was collected, but this would have no relevance to whether temperatures are higher or lower. In order to have*	Inference and Models	7. Communicating about computing

	sufficient and accurate data on which to communicate the idea, students might want to use data provided by a governmental weather agency.		

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
5.AP.A.01	Analyze and refine multiple algorithms for the same task and determine which algorithm is the most efficient. *Different algorithms can achieve the same result; however, sometimes one algorithm might be more suited for a particular situation. Students should be able to look at different ways to solve a problem or complete a task and decide which would be the best solution. For example, students could create multiple algorithms to plan a route between two points on a map. They could then look at different mapping software to change the route based on something that would be better (i.e. shortest route in miles, time, toll roads, etc.). Students could also compare algorithms that describe how to get ready for school or other daily tasks. This could also bridge into other disciplines. For example, students could write different algorithms to draw a regular polygon and determine which algorithm would be the easiest to modify or repurpose to draw a different polygon and which algorithm would be the most efficient at completing the polygon.*	Algorithms	3. Recognizing and Defining Computational Problems 6. Testing and Refining Computational Artifacts
5.AP.V.01	Utilize, create, and modify programs that use, modify, and combine variables with grade level appropriate data. *Variables are used to store and modify data. At this level, understanding how to use variables in a variety of ways and change variable values is sufficient. For example, students may use mathematical operations to add to the score of a game or subtract from the number of lives available in a game. Students could also use multiple variables in mathematical equations (even simple addition/subtraction) that stores the results into other variables. The use of a variable is a countdown timer is another example.*	Variables	5. Creating computational artifacts
5.AP.C.01	Create programs using a programming language that utilize sequencing, repetition, conditionals, event handlers, and variables to solve a problem or express ideas both independently and collaboratively.	Control	5. Creating computational artifacts

	Control structures specify the order (sequence) in which instructions are executed within a program and can be combined to support the creation of more complex programs. Events allow portions of a program to run based on a specific action. For example, students could write a program to explain the water cycle and when a specific component is clicked (event), the program would show information about that part of the water cycle. Conditionals allow for the execution of a portion of code in a program when a certain condition is true. For example, students could write a math game that asks for multiplication fact questions and then uses a conditional to check whether or not the answer that was entered is correct. Loops allow for the repetition of a sequence of code multiple times. For example, in a program that produces an animation about a famous historical character, students could use a loop to have the character walk across the screen as they introduce themselves. Students should be able to complete these tasks collaboratively with other students.		
5.AP.M.01	Decompose (break down) large problems into smaller, more manageable subproblems to facilitate the program development process. *Decomposition is the act of breaking down tasks into simpler tasks. For example, students could create program that tells a story. Students should breakdown creating the program by separating the story into different scenes. For each scene, they would select a background, place characters, and program actions.*	Modularity	3. Recognizing and defining computational problems
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. *Programs can be broken down into smaller parts, which can be incorporated into new or existing programs. For example, students could modify prewritten code from a single-player game to create a two-player game with slightly different rules, remix and add another scene to an animated story, use code to make a ball bounce from another program*	Modularity	3. Recognizing and defining computational problems 5. Creating computational artifacts

	in a new basketball game, or modify an image created by another student.		
5.AP.PD.01	Create a plan using an iterative process for the development of a program that includes others' perspectives and user preferences while solving simple problems. *Planning is an important part of the iterative process of program development. Students outline key features, time and resource constraints, and user (and others) expectations. Students should document the plan as, for example, a storyboard, flowchart, pseudocode, or story map.*	Program Development	1. Fostering an Inclusive Computing Culture 5. Creating computational artifacts
5.AP.PD.02	Use proper citations and document when ideas are borrowed and changed for their own use (e.g., using pictures created by others, using music created by others, remixing programming projects). *Intellectual property rights can vary by country but copyright laws give the creator of a work a set of rights that prevents others from copying the work and using it in ways that they may not like. Students should identify instances of remixing, when ideas are borrowed and iterated upon, and credit the original creator. Students should also consider common licenses that place limitations or restrictions on the use of computational artifacts, such as images and music downloaded from the Internet. At this stage, attribution should be written in the format required by the teacher and should always be included on any programs shared online.*	Program Development	5. Creating computational artifacts 7. Communicating about computing
5.AP.PD.03	Analyze, debug (identify/fix errors), and create a program that includes sequencing, repetition and variables in a programming language. *As students develop programs they should continuously test those programs to see that they do what was expected and fix (debug), any errors. Students should also be able to successfully debug simple errors in programs created by others.*	Program Development	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing 6. Testing and Refining Computational Artifacts

5.AP.PD.04	Take on varying roles collaborating with peers to give feedback at different stages of program development, including design and implementation. *Collaborative computing is the process of performing a computational task by working in pairs or on teams. Because it involves asking for the contributions and feedback of others, effective collaboration can lead to better outcomes than working independently. Students should take turns in different roles during program development, such as note taker, facilitator, program tester, or "driver" of the computer.*	Program Development	2. Collaborating Around Computing 7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
5.IC.C.01	Develop, test, and refine digital artifacts to improve accessibility and usability for a computing device or program. *The development and modification of computing technology are driven by people's needs and wants and can affect groups differently. Anticipating the needs and wants of diverse end users requires students to purposefully consider potential perspectives of users with different backgrounds, ability levels, points of view, and disabilities. For example, students may consider using both speech and text to convey information in a game. They may also wish to vary the types of programs they create, knowing that not everyone shares their own tastes.*	Culture	4. Developing and Using Abstractions 5. Creating Computational Artifacts 6. Testing and Refining Computational Artifacts
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. (Digital Citizenship - review of all nine components, but focused on Digital Commerce, Digital Law, and Digital Security. *The practice of appropriate online behavior derives from the identification of inappropriate behavior and the identification of what makes someone a poor digital citizen or know what not to do in order to be ethical online. Digital citizenship is described with nine categories: digital access (awareness of materials gained online and being mindful of who does/does not have access), digital commerce (awareness of*	Social Interactions	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing

	illegal and legal exchanges online e.g. illegal downloading), digital communication (communicating and collaborating properly online), digital literacy (the ability to use new technology quickly and appropriately), digital etiquette (appropriate conduct), digital law (ethical use of technology e.g. hacking information, downloading illegally, plagiarizing, creating viruses, sending spam, or stealing someone's identify), digital rights and responsibilities (knowing your rights to free speech and privacy, but handling it responsibly online) digital health and wellness (caring for your physical and psychological wellbeing online), digital security (proactive about protecting your devices and identity online e.g. data backup, use of a surge protector, virus protection).		
5.IC.SI.02	As a team, collaborate with people and resources outside of your normal space to include diverse perspectives to improve computational products. *Computing provides the possibility for collaboration and sharing of ideas and allows the benefit of diverse perspectives. For example, students could seek feedback from other groups in their class or students at another grade level. Or, with guidance from their teacher, they could use video conferencing tools or other online collaborative spaces, such as blogs, wikis, forums, or website comments to gather feedback from individuals and groups about programming projects.*	Social Interactions	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing
5.IC.H.01	Identify and explain the evolution of computing technologies that have changed the world. *New computing technology is created and existing technologies are modified for many reasons, including to increase their benefits, decrease their risks, and meet societal needs. Students, with guidance from their teacher, should discuss topics that relate to the history of technology and the changes in the world due to technology. Topics could be based on current news content, such as robotics, wireless Internet, mobile computing devices, GPS systems, wearable computing, or how social media has influenced social and political changes.*	History	1. Fostering an Inclusive Computing Culture 7. Communicating about Computing
5.IC.SLE.01	Observe intellectual property rights and give appropriate credit when using resources.	Safety, Law, & Ethics	5. Creating Computational Artifacts

	Ethical complications arise from the opportunities provided by computing. The ease of sending and receiving copies of media on the internet such as video, photos, and music, creates the opportunity for unauthorized use, such as online piracy, and disregard of copyrights. Students should consider the licenses on computational artifacts that they wish to use. For example, the license on a downloaded image rn audio file may have restrictions that prohibit modification, require attribution, or prohibit use entirely.		
5.IC.CP.01	Design a visual product depicting the connections between computer science and other fields. *Explaining the reason why of any computer task will lead students to understand how other professionals within their community might use similar tasks in their occupations. Making correlations and a purpose for tasks makes CS relevant to their lives as they age. Students make a direct correlation to a local business.*	Community Partnerships	1. Fostering an Inclusive Computing Culture 2. Collaborating Around Computing 7. Communicating about computing

Middle Grades (Grades 6-8)

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
MG.CS.D.01	Develop and implement a process to evaluate existing computing devices and recommend improvements to design based on analysis of how other users interact with the device. *The study of human–computer interaction (HCI) can improve the design of devices, including both hardware and software. Students should make recommendations for existing devices (e.g., a laptop, phone, or tablet) or design their own components or interface (e.g., create their own controllers). Teachers can guide students to consider usability through several lenses, including accessibility, ergonomics, and learnability. For example, assistive devices provide capabilities such as scanning written information and converting it to speech.*	Devices	3. Recognizing and defining computational problems
MG.CS.HS.01	Model a computing system involving multiple considerations and potential tradeoffs of software and hardware, such as functionality, cost, size, speed, accessibility, and aesthetics *Collecting and exchanging data involves input, output, storage, and processing. When possible, students should select the hardware and software components for their project designs by considering factors such as functionality, cost, size, speed, accessibility, and aesthetics. For example, components for a mobile app could include accelerometer, GPS, and speech recognition. The choice of a device that connects wirelessly through a Bluetooth connection versus a physical USB connection involves a tradeoff between mobility and the need for an additional power source for the wireless device.*	Hardware and Software	5. Creating computational artifacts
MG.CS.IO.01	Know and apply grade-level appropriate skills with input and output devices. *Students can identify and use appropriate in-put devices (i.e. mouse, keyboard, microphone, camera, scanner) and out-put devices (i.e. monitor, printer, 3d-printer, projector, robots, audio devices, VR headsets). Create accurate typed text with speed appropriate for grade level (i.e. keyboarding between 20-30 words per minute with 90%*	Input and Output	7. Communicating about computing

	accuracy). Typing words and sentences without looking at the keyboard. Access function keys and keyboard shortcuts as needed in software applications. Type at least seven pages of text into an appropriate software program in a single setting.		
MG.CS.T.01	Systematically identify, fix, and document increasingly complex software and hardware problems with computing devices and their components. *Since a computing device may interact with interconnected devices within a system, problems may not be due to the specific computing device itself but to devices connected to it. Just as pilots use checklists to troubleshoot problems with aircraft systems, students should use a similar, structured process to troubleshoot problems with computing systems and ensure that potential solutions are not overlooked. Examples of troubleshooting strategies include following a troubleshooting flow diagram, making changes to software to see if hardware will work, checking connections and settings, and swapping in working components.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
MG.NI.NCO.01	Explain protocols and their importance to data transmission; model how packets are broken down into smaller pieces and how they are delivered. *Protocols are rules that define how messages between computers are sent. They determine how quickly and securely information is transmitted across networks and the Internet, as well as how to handle errors in transmission. Students should model how data is sent using protocols to choose the fastest path, to deal with missing information, and to deliver sensitive data securely. For example, students could devise a plan for resending lost information or for interpreting a picture that has missing pieces. The priority at this grade level is understanding the purpose of protocols and how they enable secure and errorless communication. Knowledge of the details of how specific protocols work is not expected.*	Network Communication & Organization	4. Developing and using abstractions

MG.NI.C.01	Evaluate physical and digital procedures that could be implemented to protect electronic data/information; explain the impacts of hacking, ransomware, scams, fake scans, and ethical/legal concerns. *Information that is stored online is vulnerable to unwanted access. Examples of physical security measures to protect data include keeping passwords hidden, locking doors, making backup copies on external storage devices, and erasing a storage device before it is reused. Examples of digital security measures include secure router admin passwords, firewalls that limit access to private networks, and the use of a protocol such as HTTPS to ensure secure data transmission, and two-factor authentication.*	Cybersecurity	7. Communicating about computing
MG.NI.C.02	Compare the advantages and disadvantages of multiple methods of encryption to model the secure transmission of information. *Encryption can be as simple as letter substitution or as complicated as modern methods used to secure networks and the Internet. Students should encode and decode messages using a variety of encryption methods, and they should understand the different levels of complexity used to hide or secure information. For example, students could secure messages using methods such as Caesar ciphers or steganography (i.e., hiding messages inside a picture or other data). They can also model more complicated methods, such as public key encryption, through unplugged activities.*	Cybersecurity	4. Developing and using abstractions

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
MG.DA.S.01	Analyze multiple methods of representing data and choose the most appropriate method for representing data. *Data representations occur at multiple levels of abstraction, from the physical storage of bits to the arrangement of information into organized formats (e.g., tables). Students should represent the same data in multiple ways. For example, students could represent the same color using binary, RGB values, hex codes (low-level representations), as*	Storage	4. Developing and using abstractions

	well as forms understandable by people, including words, symbols, and digital displays of the color (high-level representations).		
MG.DA.C.01	Develop, implement, and refine a process that utilizes computational tools to collect meaningful data. *Students need to be able to distinguish between different types of data and computational tools and how this affects the accuracy and precision of the data (for example, surveys versus sensor data).*	Collection	6. Testing and refining computational artifacts
MG.DA.CVT.01	Develop, implement, and refine a process to make data more useful and reliable. *As students continue to build on their ability to organize and present data visually to support a claim, they will need to understand when and how to transform data for this purpose. Students should transform data to remove errors, highlight or expose relationships, and/or make it easier for computers to process. The cleaning of data is an important transformation for ensuring consistent format and reducing noise and errors (e.g., removing irrelevant responses in a survey). An example of a transformation that highlights a relationship is representing males and females as percentages of a whole instead of as individual counts.*	Visualization & Transformation	6. Testing and refining computational artifacts
MG.DA.IM.01	Refine computational models based on the data generated by the models. *A model may be a programmed simulation of events or a representation of how various data is related. In order to refine a model, students need to consider which data points are relevant, how data points relate to each other, and if the data is accurate. For example, students may make a prediction about how far a ball will travel based on a table of data related to the height and angle of a track. The students could then test and refine their model by comparing predicted versus actual results and considering whether other factors are relevant (e.g., size and mass of the ball). Additionally, students could refine game mechanics based on test outcomes in order to make the game more balanced or fair.*	Inference and Models	4. Developing and using abstractions 5. Creating computational artifacts

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
MG.AP.A.01	Design algorithms in natural language, flow and control diagrams, comments within code, and/or pseudocode to solve complex problems. *Complex problems are problems that would be difficult for students to solve computationally. Students should use pseudocode and/or flowcharts to organize and sequence an algorithm that addresses a complex problem, even though they may not actually program the solutions. For example, students might express an algorithm that produces a recommendation for purchasing sneakers based on inputs such as size, colors, brand, comfort, and cost. Testing the algorithm with a wide range of inputs and users allows students to refine their recommendation algorithm and to identify other inputs they may have initially excluded.*	Algorithms	4. Developing and using abstractions
MG.AP.V.01	Create programs using variables with purposeful and thoughtful naming conventions for identifiers to improve program readability. *A variable is like a container with a name, in which the contents may change, but the name (identifier) does not. When planning and developing programs, students should decide when and how to declare and name new variables. Students should use naming conventions to improve program readability. Examples of operations include adding points to the score, combining user input with words to make a sentence, changing the size of a picture, or adding a name to a list of people.*	Variables	5. Creating computational artifacts
MG.AP.C.01	Develop programs that utilize combinations of nested repetition, compound conditionals, procedures without parameters, and the manipulation of variables representing different data types. *Control structures can be combined in many ways. Nested loops are loops placed within loops. Compound conditionals combine two or more conditions in a logical relationship (e.g., using AND, OR, and NOT), and nesting conditionals within one another allows the result of one conditional to lead to another. For example, when programming an interactive story, students could use a compound conditional within a*	Control	5. Creating computational artifacts

	loop to unlock a door only if a character has a key AND is touching the door.		
MG.AP.M.01	Decompose problems and subproblems into parts to facilitate the design, implementation, and review of complex programs. *Students should break down problems into subproblems, which can be further broken down to smaller parts. Decomposition facilitates aspects of program development by allowing students to focus on one piece at a time (e.g., getting input from the user, processing the data, and displaying the result to the user). Decomposition also enables different students to work on different parts at the same time. For example, animations can be decomposed into multiple scenes, which can be developed independently.*	Modularity	3. Recognizing and defining computational problems
MG.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. *Development teams that employ user-centered design create solutions (e.g., programs and devices) that can have a large societal impact, such as an app that allows people with speech difficulties to translate hard-to-understand pronunciation into understandable language. Students should begin to seek diverse perspectives throughout the design process to improve their computational artifacts. Considerations of the end-user may include usability, accessibility, age-appropriate content, respectful language, user perspective, pronoun use, color contrast, and ease of use.*	Program Development	1. Fostering an inclusive computing culture 2. Collaborating around computing
MG.AP.PD.02	Incorporate existing code, media, and libraries into original programs of increasing complexity and give attribution. *Building on the work of others enables students to produce more interesting and powerful creations. Students should use portions of code, algorithms, and/or digital media in their own programs and websites. At this level, they may also import libraries and connect to web application program interfaces (APIs). For example, when creating a side-scrolling game, students may incorporate portions of code that create a realistic jump movement from another person's game, and they may also import Creative Commons-licensed images to use in the*	Program Development	4. Developing and using abstractions 5. Creating computational artifacts 7. Communicating about computing

	background. Students should give attribution to the original creators to acknowledge their contributions.		
MG.AP.PD.03	Systematically test and refine programs using a range of student created inputs. *Use cases and test cases are created and analyzed to better meet the needs of users and to evaluate whether programs function as intended. At this level, testing should become a deliberate process that is more iterative, systematic, and proactive than at lower levels. Students should begin to test programs by considering potential errors, such as what will happen if a user enters invalid input (e.g., negative numbers and 0 instead of positive numbers).*	Program Development	6. Testing and refining computational artifacts
MG.AP.PD.04	Explain how effective communication between participants is required for successful collaboration when developing computational artifacts. *Collaboration is a common and crucial practice in programming development. Often, many individuals and groups work on the interdependent parts of a project together. Students should assume pre-defined roles within their teams and manage the project workflow using structured timelines. With teacher guidance, they will begin to create collective goals, expectations, and equitable workloads. For example, students may divide the design stage of a game into planning the storyboard, flowchart, and different parts of the game mechanics. They can then distribute tasks and roles among members of the team, assign deadlines, and track progress towards goals.*	Program Development	2. Collaborating around computing
MG.AP.PD.05	Document text-based programs of increasing complexity in order to make them easier to follow, test, and debug. *Documentation allows creators and others to more easily use and understand a program. Students should provide documentation for end users that explains their artifacts and how they function. For example, students could provide a project overview and clear user instructions. They should also incorporate comments in their product and communicate their process using design documents, flowcharts, and presentations.*	Program Development	7. Communicating about Computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
MG.IC.C.01	Describe the trade-offs associated with computing technologies (e.g. automation), explaining their effects on economies and global societies, and explore careers related to the field of computer science. *Advancements in computer technology are neither wholly positive nor negative. However, the ways that people use computing technologies have tradeoffs. Students should consider current events related to broad ideas, including privacy, communication, and automation. For example, driverless cars can increase convenience and reduce accidents, but they are also susceptible to hacking. The emerging industry will reduce the number of taxi and shared-ride drivers, but will create more software engineering and cybersecurity jobs.*	Culture	7. Communicating about computing
MG.IC.C.02	Evaluate and improve the design of existing technologies to meet the needs of diverse users and increase accessibility and usability. *Students should test and discuss the usability of various technology tools (e.g., apps, games, and devices) with the teacher's guidance. For example, facial recognition software that works better for lighter skin tones was likely developed with a homogeneous testing group and could be improved by sampling a more diverse population. When discussing accessibility, students may notice that allowing a user to change font sizes and colors will not only make an interface usable for people with low vision but also benefits users in various situations, such as in bright daylight or a dark room.*	Culture	1. Fostering an inclusive computing culture
MG.IC.SI.01	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. Describe and use safe, appropriate, and responsible practices (netiquette) when participating in online communities (e.g., discussion groups, blogs, social networking sites). *Crowdsourcing is gathering services, ideas, or content from a large group of people, especially from the online community. It can be done at the local level (e.g., classroom or school) or global level (e.g., age*	Social Interactions	2. Collaborating around computing 5. Creating computational artifacts

	appropriate online communities, like Scratch and Minecraft). For example, a group of students could combine animations to create a digital community mosaic. They could also solicit feedback from many people though use of online communities and electronic surveys..		
MG.IC.H.01	Identify and describe how the prominent figures in computer science have impacted and/or progressed the field. *Students will identify and understand how prominent figures in computer science (i.e. Charles Babbage, Alan Turning, Ada Lovelace, Bill Gates, Tim Berners-Lee) impacted growth and innovation in the field of Computer Science.*	History	3. Recognizing and defining computational problems
MG.IC.SLE.01	Discuss the social impacts and ethical considerations associated with cybersecurity, including the positive and malicious purposes of hacking. *Sharing information online can help establish, maintain, and strengthen connections between people. For example, it allows artists and designers to display their talents and reach a broad audience. However, security attacks often start with personal information that is publicly available online. Social engineering is based on tricking people into revealing sensitive information and can be thwarted by being wary of attacks, such as phishing and spoofing.*	Safety, Law, & Ethics	7. Communicating about computing
MG.IC.CP.01	Formulate a computer-science based solution for a problem or issue by gathering input from local / regional industry members. *Students will work with local / regional community members to identify and address a need using computer science practices.*	Community Partnerships	2. Collaborating around computing 5. Creating computational artifacts

Secondary Grades L1 (Grades 9-12) (All Students)

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L1.CS.D.01	Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. *Computing devices are often integrated with other systems, including biological, mechanical, and social systems. A medical device can be embedded inside a person to monitor and regulate his or her health, a hearing aid (a type of assistive device) can filter out certain frequencies and magnify others, a monitoring device installed in a motor vehicle can track a person's driving patterns and habits, and a facial recognition device can be integrated into a security system to identify a person. The creation of integrated or embedded systems is not an expectation at this level. Students might select an embedded device such as a car stereo, identify the types of data (radio station presets, volume level) and procedures (increase volume, store/recall saved station, mute) it includes, and explain how the implementation details are hidden from the user.*	Devices	4. Developing and using abstractions
L1.CS.HS.01	Compare levels of abstraction and interactions between application software, system software, and hardware layers. *At its most basic level, a computer is composed of physical hardware and electrical impulses. Multiple layers of software are built upon the hardware and interact with the layers above and below them to reduce complexity. System software manages a computing device's resources so that software can interact with hardware. For example, text editing software interacts with the operating system to receive input from the keyboard, convert the input to bits for storage, and interpret the bits as readable text to display on the monitor. System software is used on many different types of devices, such as smart TVs, assistive devices, virtual components, cloud components, and drones. For example, students may explore the progression from voltage to binary signal to logic gates to adders and so on. Knowledge of specific, advanced terms for computer architecture, such as BIOS, kernel, or bus, is not expected at this level.*	Hardware and Software	4. Developing and using abstractions

L1.CS.HS.02	Compare computer systems and determine advantages and drawbacks of each system.	Hardware and Software	7. Communicating about computing
L1.CS.IO.01	Demonstrate efficient use of input and output devices. *Fluency in educational and industry specific input and output devices including keyboard, mouse, touch screen, microphone, speakers, screen representations, printing, and other specific input and output devices.*	Input and Output	7. Communicating about computing
L1.CS.T.01	Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. *Troubleshooting complex problems involves the use of multiple sources when researching, evaluating, and implementing potential solutions. Troubleshooting also relies on experience, such as when people recognize that a problem is similar to one they have seen before or adapt solutions that have worked in the past. Examples of complex troubleshooting strategies include resolving connectivity problems, adjusting system configurations and settings, ensuring hardware and software compatibility, and transferring data from one device to another. Students could create a flow chart, a job aid for a help desk employee, or an expert system.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L1.NI.NCO.01	Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. *Each device is assigned an address that uniquely identifies it on the network. Routers function by comparing IP addresses to determine the pathways packets should take to reach their destination. Switches function by comparing MAC addresses to determine which computers or network segments will receive frames. Students could use online network simulators to experiment with these factors.*	Network Communication & Organization	4. Developing and using abstractions

L1.NI.NCO.02	Compare various security measures, considering tradeoffs between the usability and security of a computing system. *Security measures may include physical security tokens, two-factor authentication, and biometric verification, but choosing security measures involves tradeoffs between the usability and security of the system. The needs of users and the sensitivity of data determine the level of security implemented. Students might discuss computer security policies in place at the local level that present a tradeoff between usability and security, such as a web filter that prevents access to many educational sites but keeps the campus network safe.*	Network Communication & Organization	6. Testing and refining computational artifacts
L1.NI.C.01	Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts. *Security measures may include physical security tokens, two-factor authentication, and biometric verification. Potential security problems, such as denial-of-service attacks, ransomware, viruses, worms, spyware, and phishing, exemplify why sensitive data should be securely stored and transmitted. The timely and reliable access to data and information services by authorized users, referred to as availability, is ensured through adequate bandwidth, backups, and other measures. Students should systematically evaluate the feasibility of using computational tools to solve given problems or subproblems, such as through a cost-benefit analysis. Eventually, students should include more factors in their evaluations, such as how efficiency affects feasibility or whether a proposed approach raises ethical concerns.*	Cybersecurity	3. Recognizing and defining computational problems
L1.NI.C.02	Explain tradeoffs when selecting and implementing cybersecurity recommendations. *Network security depends on a combination of hardware, software, and practices that control access to data and systems. The needs of users and the sensitivity of data determine the level of security implemented. Every security measure involves tradeoffs between the accessibility and security of the system. Students should be able to describe, justify, and document choices they make using terminology appropriate for the intended audience and purpose. Students could debate issues from the*	Cybersecurity	7. Communicating about computing

	perspective of diverse audiences, including individuals, corporations, privacy advocates, security experts, and government.		

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L1.DA.S.01	Analyze storage types and locations.	Storage	4. Developing and using abstractions
L1.DA.S.02	Evaluate the tradeoffs in how data elements are organized and where data is stored. *People make choices about how data elements are organized and where data is stored. These choices affect cost, speed, reliability, accessibility, privacy, and integrity. Students should evaluate whether a chosen solution is most appropriate for a particular problem. Students might consider the cost, speed, reliability, accessibility, privacy, and integrity tradeoffs between storing photo data on a mobile device versus in the cloud.*	Storage	3. Recognizing and defining computational problems
L1.DA.C.01	Collect and analyze data.	Collection	4. Developing and using abstractions
L1.DA.CVT.01	Create interactive data visualizations using software tools to help others better understand real-world phenomena. *People transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information. Examples include visualization, aggregation, rearrangement, and application of mathematical operations. People use software tools or programming to create powerful, interactive data visualizations and perform a range of mathematical operations to transform and analyze data. Students should model phenomena as systems, with rules governing the interactions within the system and evaluate these models against real-world observations. For example, flocking behaviors, queueing, or life cycles. Google Fusion Tables can provide access to data visualization online.*	Visualization & Transformation	4. Developing and using abstractions

L1.DA.IM.01	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process. *Computational models make predictions about processes or phenomenon based on selected data and features. The amount, quality, and diversity of data and the features chosen can affect the quality of a model and ability to understand a system. Predictions or inferences are tested to validate models. Students should model phenomena as systems, with rules governing the interactions within the system. Students should analyze and evaluate these models against real-world observations. For example, students might create a simple producer–consumer ecosystem model using a programming tool. Eventually, they could progress to creating more complex and realistic interactions between species, such as predation, competition, or symbiosis, and evaluate the model based on data gathered from nature.*	Inference and Models	4. Developing and using abstractions

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L1.AP.A.01	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. *A prototype is a computational artifact that demonstrates the core functionality of a product or process. Prototypes are useful for getting early feedback in the design process, and can yield insight into the feasibility of a product. 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. Students should develop artifacts in response to a task or a computational problem that demonstrate the performance, reusability, and ease of implementation of an algorithm.*	Algorithms	5. Creating computational artifacts
L1.AP.V.01	Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.	Variables	4. Developing and using abstractions

	Students should be able to identify common features in multiple segments of code and substitute a single segment that uses lists (arrays) to account for the differences.		
L1.AP.C.01	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made. *Implementation includes the choice of programming language, which affects the time and effort required to create a program. Readability refers to how clear the program is to other programmers and can be improved through documentation. The discussion of performance is limited to a theoretical understanding of execution time and storage requirements; a quantitative analysis is not expected. Control structures at this level may include conditional statements, loops, event handlers, and recursion. For example, students might compare the readability and program performance of iterative and recursive implementations of procedures that calculate the Fibonacci sequence.*	Control	5. Creating computational artifacts
L1.AP.C.02	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. *In this context, relevant computational artifacts include programs, mobile apps, or web apps. Events can be user-initiated, such as a button press, or system-initiated, such as a timer firing. At previous levels, students have learned to create and call procedures. Here, students design procedures that are called by events. Students might create a mobile app that updates a list of nearby points of interest when the device detects that its location has been changed.*	Control	5. Creating computational artifacts
L1.AP.C.03	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. *At this level, students should decompose complex problems into manageable subproblems that could potentially be solved with programs or procedures that already exist. For example, students could create an app to solve a community problem by connecting to an online database through an application programming interface (API).*	Control	5. Creating computational artifacts

L1.AP.M.01	Create computational artifacts by systematically organizing, manipulating and/or processing data. *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. Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. Modules allow for better management of complex tasks. The focus at this level is understanding a program as a system with relationships between modules. The choice of implementation, such as programming language or paradigm, may vary. Students could incorporate computer vision libraries to increase the capabilities of a robot or leverage open-source JavaScript libraries to expand the functionality of a web application.*	Modularity	3. Recognizing and defining computational problems
L1.AP.M.02	Systematically design and develop programs for broad audiences by incorporating feedback from users. *Examples of programs could include games, utilities, and mobile applications. Students at lower levels collect feedback and revise programs. At this level, students should do so through a systematic process that includes feedback from broad audiences. Students might create a user satisfaction survey and brainstorm distribution methods that could yield feedback from a diverse audience, documenting the process they took to incorporate selected feedback in product revisions.*	Modularity	5. Creating computational artifacts
L1.AP.PD.01	Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. *Examples of software licenses include copyright, freeware, and the many open-source licensing schemes. At previous levels, students adhered to licensing schemes. At this level, they should consider licensing implications for their own work, especially when incorporating libraries and other resources. Students might consider two software libraries that address a similar need, justifying their choice based on the library that has the least restrictive license.*	Program Development	7. Communicating about computing

L1.AP.PD.02	Evaluate and refine computational artifacts to make them more usable and accessible. *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 should respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts. For example, students could incorporate feedback from a variety of end users to help guide the size and placement of menus and buttons in a user interface.*	Program Development	6. Testing and refining computational artifacts
L1.AP.PD.03	Design and develop computational artifacts working in team roles using collaborative tools. *Collaborative tools could be as complex as source code version control system or as simple as a collaborative word processor. Team roles in pair programming are driver and navigator but could be more specialized in larger teams. As programs grow more complex, the choice of resources that aid program development becomes increasingly important and should be made by the students. Students might work as a team to develop a mobile application that addresses a problem relevant to the school or community, selecting appropriate tools to establish and manage the project timeline; design, share, and revise graphical user interface elements; and track planned, in-progress, and completed components.*	Program Development	2. Collaborating around computing
L1.AP.PD.04	Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. *Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. These modules can be procedures within a program; combinations of data and procedures; or independent, but interrelated, programs. The development of complex programs is aided by resources such as libraries and tools to edit and manage parts of the program.*	Program Development	7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L1.IC.C.01	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices. *Computing may improve, harm, or maintain practices. Equity deficits, such as minimal exposure to computing, access to education, and training opportunities, are related to larger, systemic problems in society. Students should be able to evaluate the accessibility of a product to a broad group of end users, such as people who lack access to broadband or who have various disabilities. Students should also begin to identify potential bias during the design process to maximize accessibility in product design.*	Culture	1. Fostering an inclusive computing culture
L1.IC.C.02	Test and refine computational artifacts to reduce bias and equity deficits. *Biases could include incorrect assumptions developers have made about their user base. Equity deficits include minimal exposure to computing, access to education, and training opportunities. Students should begin to identify potential bias during the design process to maximize accessibility in product design and become aware of professionally accepted accessibility standards to evaluate computational artifacts for accessibility.*	Culture	1. Fostering an inclusive computing culture
L1.IC.C.03	Demonstrate how a given algorithm applies to problems across disciplines. *Computation can share features with disciplines such as art and music by algorithmically translating human intention into an artifact. Students should be able to identify real-world problems that span multiple disciplines, such as increasing bike safety with new helmet technology, and that can be solved computationally.*	Culture	3. Recognizing and defining computational problems
L1.IC.SI.01	Compare and contrast the benefits and drawbacks of social media.	Social Interactions	2. Collaborating around computing
L1.IC.H.01	Hypothesize the impact of the innovations of computing systems for the next decade.	History	7. Communicating about computing

	As computers become interconnected in each aspect of society, more powerful, and students become more reliant on them, students should be able describe the number of times computers or devices are accessed each day by teachers or peers in class and discuss what life would be like without them.		
L1.IC.SLE.01	Explain the beneficial and harmful effects that intellectual property laws can have on innovation. *Laws govern many aspects of computing, such as privacy, data, property, information, and identity. These laws can have beneficial and harmful effects, such as expediting or delaying advancements in computing and protecting or infringing upon people's rights. International differences in laws and ethics have implications for computing. For examples, laws that mandate the blocking of some file-sharing websites may reduce online piracy but can restrict the right to access information. Firewalls can be used to block harmful viruses and malware but can also be used for media censorship. Students should be aware of intellectual property laws and be able to explain how they are used to protect the interests of innovators and how patent trolls abuse the laws for financial gain.*	Safety, Law, & Ethics	7. Communicating about computing
L1.IC.SLE.02	Explain the privacy concerns related to the collection and generation of data through automated processes (e.g., how businesses, social media, and the government collects and uses data) that may not be evident to users. *Data can be collected and aggregated across millions of people, even when they are not actively engaging with or physically near the data collection devices. This automated and nonevident collection can raise privacy concerns, such as social media sites mining an account even when the user is not online. Other examples include surveillance video used in a store to track customers for security or information about purchase habits or the monitoring of road traffic to change signals in real time to improve road efficiency without drivers being aware. Methods and devices for collecting data can differ by the amount of storage required, level of detail collected, and sampling rates.*	Safety, Law, & Ethics	7. Communicating about computing

L1.IC.SLE.03	Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. *Laws govern many aspects of computing, such as privacy, data, property, information, and identity. International differences in laws and ethics have implications for computing. Students might review case studies or current events which present an ethical dilemma when an individual's right to privacy is at odds with the safety, security, or wellbeing of a community.*	Safety, Law, & Ethics	7. Communicating about computing
L1.IC.CP.01	Explore computing, software, and data storage systems in local industries.	Community Partnerships	7. Communicating about computing

Secondary Grades L2 (Grades 9-12) (Students who wish to pursue computer science beyond what is expected of all students)

Computing Systems

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L2.CS.D.01	Describe how internal and external parts of computing devices function to form a system.	Devices	4. Developing and using abstractions
L2.CS.HS.01	Categorize the roles of operating system software. *Examples of roles could include memory management, data storage/retrieval, processes management, and access control.*	Hardware and Software	4. Developing and using abstractions
L2.CS.HS.02	Compare options for building a computer systems and determine advantages and drawbacks of each piece and how it will affect the overall performance.	Hardware and Software	7. Communicating about computing
L2.CS.IO.01	Demonstrate use of course specific advanced input and output devices related to field. *Examples could include robotics, joysticks, motion sensors, movement sensors, GPS, and various other specific to CTE courses.*	Input and Output	7. Communicating about computing
L2.CS.T.01	Illustrate ways computing systems implement logic, input, and output through hardware components. *Examples of components could include logic gates and IO pins.*	Troubleshooting	6. Testing and refining computational artifacts

Networks & the Internet

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L2.NI.NCO.01	Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology). *Recommend use of free online network simulators to explore how these issues impact network functionality.*	Network Communication & Organization	4. Developing and using abstractions

L2.NI.NCO.02	Give examples to illustrate how sensitive data can be affected by malware and other attacks.	Network Communication & Organization	6. Testing and refining computational artifacts
L2.NI.C.01	Compare ways software developers protect devices and information from unauthorized access. *Examples of security concerns to consider: encryption and authentication strategies, secure coding, and safeguarding keys.*	Cybersecurity	3. Recognizing and defining computational problems
L2.NI.C.02	Use encryption and decryption algorithms to transmit/ receive an encrypted message.	Cybersecurity	7. Communicating about computing

Data Analysis

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L2.DA.S.01	Translate and compare different bit representations of data types, such as characters, numbers, and images.	Storage	4. Developing and using abstractions
L2.DA.S.02	Analyze file systems created for keeping track of files on the hard disk.	Storage	3. Recognizing and defining computational problems
L2.DA.C.01	Select data collection tools and techniques to generate data sets that support a claim or communicate information.	Collection	4. Developing and using abstractions
L2.DA.CVT.01	Use data analysis tools and techniques to identify patterns in data representing complex systems. *For example, identify trends in a dataset representing social media interactions, movie reviews, or shopping patterns.*	Visualization & Transformation	4. Developing and using abstractions
L2.DA.IM.01	Evaluate the ability of models and simulations to test and support the refinement of hypotheses. (e.g., flocking behaviors, life cycles, etc.)	Inference and Models	4. Developing and using abstractions

Algorithms and Programming

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L2.AP.A.01	Describe how artificial intelligence algorithms drive many software and physical systems (e.g., digital advertising, autonomous robots, computer vision, pattern recognition, text analysis).	Algorithms	5. Creating computational artifacts
L2.AP.A.02	Describe how artificial intelligence drives many software and physical systems. *Examples include digital ad delivery, self-driving cars, and credit card fraud detection.*	Algorithms	5. Creating computational artifacts
L2.AP.A.03	Critically examine and trace classic algorithms (e.g., selection sort, insertion sort, binary search, linear search).	Algorithms	5. Creating computational artifacts
L2.AP.A.04	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem. *Games do not have to be complex. Simple guessing games, Tic-Tac-Toe, or simple robot commands will be sufficient.*	Algorithms	5. Creating computational artifacts
L2.AP.A.05	Use and adapt classic algorithms to solve computational problems. *Examples could include sorting and searching.*	Algorithms	5. Creating computational artifacts
L2.AP.A.06	Evaluate algorithms in terms of their efficiency, correctness, and clarity. *Examples could include sorting and searching.*	Algorithms	5. Creating computational artifacts
L2.AP.V.01	Compare and contrast simple data structures and their uses to simplify solutions, generalizing computational problems instead of repeatedly using primitive variables. *Examples could include strings, lists, arrays, stacks, and queues.*	Variables	4. Developing and using abstractions
L2.AP.C.01	Trace the execution of repetition (e.g., loops, recursion), illustrating output and changes in values of named variables.	Control	5. Creating computational artifacts

L2.AP.M.01	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.	Modularity	3. Recognizing and defining computational problems
L2.AP.M.02	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. *As students encounter complex, real-world problems that span multiple disciplines or social systems, they should decompose complex problems into manageable subproblems that could potentially be solved with programs or procedures that already exist. For example, students could create an app to solve a community problem by connecting to an online database through an application programming interface (API).*	Modularity	5. Creating computational artifacts
L2.AP.M.03	Demonstrate code reuse by creating programming solutions using libraries and APIs. *Libraries and APIs can be student-created or common graphics libraries or maps APIs, for example.*	Modularity	5. Creating computational artifacts
L2.AP.PD.01	Plan and develop programs for broad audiences using a software life cycle process. *Processes could include agile, spiral, or waterfall.*	Program Development	7. Communicating about computing
L2.AP.PD.02	Explain security issues that might lead to compromised computer programs. *For example, common issues include lack of bounds checking, poor input validation, and circular references.*	Program Development	6. Testing and refining computational artifacts
L2.AP.PD.03	Develop programs for multiple computing platforms. *Example platforms could include: computer desktop, web, or mobile.*	Program Development	2. Collaborating around computing
L2.AP.PD.04	Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project. *Group software projects can be assigned or student-selected.*	Program Development	7. Communicating about computing

L2.AP.PD.05	Develop and use a series of test cases to verify that a program performs according to its design specifications. *At this level, students are expected to select their own test cases.*	Program Development	7. Communicating about computing
L2.AP.PD.06	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality). *For instance, changes made to a method or function signature could break invocations of that method elsewhere in a system.*	Program Development	7. Communicating about computing
L2.AP.PD.07	Evaluate key qualities of a program through a process such as a code review. *Examples of qualities could include correctness, usability, readability, efficiency, portability and scalability.*	Program Development	7. Communicating about computing
L2.AP.PD.08	Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems. *Examples of features include blocks versus text, indentation versus curly braces, and high-level versus low- level.*	Program Development	7. Communicating about computing

Impacts of Computing

Identifier	Standard and Descriptive Statement	Subconcept	Practice(s)
L2.IC.C.01	Evaluate the beneficial and harmful effects that computational artifacts and innovations have on society.	Culture	1. Fostering an inclusive computing culture
L2.IC.C.02	Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.	Culture	1. Fostering an inclusive computing culture
L2.IC.C.03	Design and implement a study that evaluates or predicts how computing has revolutionized an aspect of our culture and how it might evolve (e.g., education, healthcare, art/entertainment, energy). *Areas to consider might include education, healthcare, art/entertainment, and energy.*	Culture	3. Recognizing and defining computational problems

L2.IC.SI.01	Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.	Social Interactions	2. Collaborating around computing
L2.IC.H.01	Analyze trends of computing and how those trends have changed over time.	History	7. Communicating about computing
L2.IC.SLE.01	Debate laws and regulations that impact the development and use of software.	Safety, Law, & Ethics	7. Communicating about computing
L2.IC.SLE.02	Determine ways to test the validity of information located online.	Safety, Law, & Ethics	7. Communicating about computing
L2.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.	Safety, Law, & Ethics	7. Communicating about computing
L2.IC.CP.01	Collaborate with local industry partners to design and implement a viable mentorship.	Community Partnerships	2. Collaborating around computing