Public Schools of North Carolina
State Board of Education | Department of Public Instruction

2020 North Carolina K12 Computer Science Standards with Descriptions.

This document is designed to help North Carolina educators teach the NC Standard Course of Study for Computer Science.

This document provides more detailed descriptions of each standard in the 2020 NC K12 Computer Science Standards which are based on the 2017 Computer Science Teachers Association Computer Science Standards.

Grades Third through Fifth

35-CS-01	**Evaluate the features available on digital devices to perform a variety of classroom tasks.**	**Computing Systems** Devices
	As students are exposed to more and varied devices, they should begin to recognize the strengths and weaknesses of these devices. Students should be able to determine the appropriateness of a device for a given task based on its features. For example, which devices might be ideal for drawing, capturing video or audio, and editing media. Students should also understand how to use different devices to complete a given task, where some work is done on a simple device and then moved to a more capable device.	
35-CS-02	**Model how computer hardware and software work together as a system to accomplish tasks.**	**Computing Systems** Hardware & Software
	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 the 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, program an animation to demonstrate it, or demonstrate it by acting this out in some way.	
35-CS-03	**Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies.**	**Computing Systems** Troubleshooting
	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 errors occur at school, the goal would be that students would use various strategies, such as rebooting the device, checking for power, checking network availability, closing and reopening an app, making sure speakers are turned on or headphones are plugged in, and making sure that the caps lock key is not on, to solve these problems, when possible.	
35-NI-01	**Model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination.**	**Networks & the Internet** Network Communication & Organization
	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 by, for instance, drawing a model of the way packets are transmitted, programming an animation to show how packets are transmitted, or demonstrating this through an unplugged activity which has them act it out in some way.	

35-NI-02	**Explain your digital footprint and how personal information can be protected.**	**Networks & the Internet** Cybersecurity
	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 data security issues. Discussion topics could be based on current events related to cybersecurity or topics that are applicable to students, such as the necessity of 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.	
35-DA-01	**Identify the type of data encoded in a file based on file extension.**	**Data & Analysis** Storage
	Before information can be stored on a computing device, the information must be converted, encoded. In many computing systems, the way information is stored depends on the type of information (text, images, movies, ...). In many cases, you can determine the type of information from the file name extension (.DOC, .GIF, .MOV) Teachers could use word walls or other classroom decorations and label them with file extensions.	
35-DA-02	**Illustrate the process of file management and version control.**	**Data & Analysis** Storage
	As we work in an ever more digital world, managing digital clutter is critical. Students should understand how files are stored on various devices and using various operating systems. In addition, students should manage progress on projects using a system to indicate versions and iterations.	
35-DA-03	**Organize and present collected data visually to highlight relationships and support a claim.**	**Data & Analysis** Collection Visualization & Transformation
	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.	
35-DA-04	**Communicate using data to highlight or predict outcomes.**	**Data & Analysis**

		Inference & Models
	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.	
35-AP-01	**Create multiple algorithms for the same task to determine which is the most accurate and efficient.**	**Algorithms & Programming** Algorithms
	Different algorithms can achieve the same result, though sometimes one algorithm might be most appropriate for a specific situation. Students should be able to look at different ways to solve the same task and decide which would be the best solution. For example, students could use a map and plan multiple algorithms to get from one point to another. They could look at routes suggested by mapping software and change the route to something that would be better, based on which route is shortest or fastest or would avoid a problem. Students might compare algorithms that describe how to get ready for school. Another example might be to 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.	
35-AP-02	**Create programs that use variables to store and modify data.**	**Algorithms & Programming** Variables
	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 as a countdown timer is another example.	
35-AP-03	**Construct programs that include sequences.**	**Algorithms & Programming** Control
	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.	

35-AP-04	**Construct programs using simple loops.**	**Algorithms & Programming** Control
	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.	
35-AP-05	**Construct programs that implement conditionals.**	**Algorithms & Programming** Control
	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 multiplication fact questions and then uses a conditional to check whether or not the answer that was entered is correct.	
35-AP-06	**Decompose problems into smaller, manageable, subproblems to facilitate the program development process.**	**Algorithms & Programming** Modularity
	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 program actions.	
35-AP-07	**Modify, remix, or incorporate portions of an existing program into one's own work.**	**Algorithms & Programming** Modularity
	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.	
35-AP-08	**Apply an iterative process to the development of a program by including diverse perspectives and considering user preferences.**	**Algorithms & Programming** Program Development
	Planning is an important part of the iterative process of program development. Students outline key features, time and resource constraints, and user expectations. Students should document the plan as, for example, a storyboard, flowchart, pseudocode, or story map.	
35-AP-09	**Give appropriate attribution when creating or remixing programs while respecting intellectual property rights.**	**Algorithms & Programming** Program Development

	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.	
35-AP-10	**Identify and debug errors in an algorithm or program to ensure it runs as intended.**	**Algorithms & Programming** Program Development
	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.	
35-AP-11	**Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development.**	**Algorithms & Programming** Program Development
	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.	
35-AP-12	**Describe choices made during program development using code comments, presentations, and demonstrations.**	**Algorithms & Programming** Program Development
	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.	
35-IC-01	**Compare computing technologies that have changed the world and how they both influence and are influenced by cultural practices.**	**Impacts of Computing** Culture
	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. Students should compare the relative impact of various innovations.	

35-IC-02	**Explore the tools that can be used to improve accessibility and usability of technology products for the diverse needs and wants of users.**	**Impacts of Computing** Culture
	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 when they wish 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.	
35-IC-03	**Seek diverse perspectives with collaboration for the purpose of improving computational artifacts.**	**Impacts of Computing** Social Interactions
	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.	
35-IC-04	**Exhibit positive digital citizenship and social responsibility.**	**Impacts of Computing** Social Interactions
	Students should recognize that the digital world is shared, and as with the physical world we inhabit, we each have a role in keeping the world safe and healthy.	
35-IC-05	**Utilize public domain or creative commons media, and refrain from copying or using material created by others without permission.**	**Impacts of Computing** Safety Law & Ethics
	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 or audio file may have restrictions that prohibit modification, require attribution, or prohibit use entirely.	