Tennessee K-12 Computer Science State Standards

K-8
Computer Science Standards

Foundational Concepts and Operations (FCO)

Computer Science Standards	Grade Level					Tennessee Academic Standards Connections Examples
	K	1-2	3-4	5-6	7-8	
FCO.1 Demonstrate fundamental technology skills (e.g., turn on and login to device).	I	I	R	M	M	• Mathematics: MP1, MP5 • Science Crosscutting Concepts: Pattern; Cause and effect
FCO.2 Interact with a device using a pointing tool such as a mouse, tactile sensor, or other input.	I	R	R	M	M	• Mathematics: MP1, MP5 • Science Crosscutting Concept: Structure and function
FCO.3 Navigate to applications and documents by using desktop icons, windows, and menus.	I	I	R	M	M	• Mathematics: MP1, MP5 • Science & Engineering Practice: Using mathematics and computational thinking
FCO.4 Use age-appropriate online tools and resources (e.g., tutorial, assessment, web browser).	I	I	R	M	M	• Mathematics: MP1, MP5 • Science & Engineering Practice: Obtaining, evaluating and communicating information
FCO.5 Demonstrate fundamental keyboarding skills.	I	I	R	R	M	• ELA: FL.PC.1 • Mathematics: MP1, MP5
FCO.6 Select and use appropriate word processing, spreadsheets, and multimedia applications.	I	I	R	R	M	• Mathematics: MP1, MP4, MP5, MP8 • Science & Engineering Practice: Using mathematics and computational thinking
FCO.7 Use menu, tool bar, and editing functions (e.g., font/size/style/line spacing, margins, spell check) to format, edit, save, and print a document.	I	I	R	R	M	• ELA: FL.WC.4 • Mathematics: MP5, MP6
FCO.8 Identify and solve routine hardware and software problems that occur during routine usage.	I	I	R	M	M	• Mathematics: MP1, MP4 • Science & Engineering Practices: Planning and carrying out controlled investigations; Constructing explanations and designing solutions • Science Crosscutting Concepts: Pattern; Cause and effect

Communication and Collaboration (CC)

Computer Science Standards	Grade Level					Tennessee Academic Standards Connection Examples
	K	1-2	3-4	5-6	7-8	
CC.1 Interact with peers, experts, and others using a variety of digital tools and devices.	I	I	R	M	M	• ELA: W.PDW.6 • Mathematics: MP3, MP6, Literacy Skills for Mathematical Proficiency • Science & Engineering Practices: Obtaining, evaluating and communicating information; Engaging in argument from evidence; Constructing explanations and designing solutions • Social Studies: SSP.01
CC.2 Communicate information and ideas effectively to multiple audiences using a variety of media and formats. (e.g., reports, research papers, presentations, newsletters, Web sites, podcasts, blogs), citing sources.	I	I	R	R	M	• ELA: SL.PKI.4, SL.PKI.5, R.RI.IKI.7 • Mathematics: MP3, MP6, Literacy Skills for Mathematical Proficiency • Science & Engineering Practice: Obtaining, evaluating and communicating information • Social Studies: SSP.01, SSP.04
CC.3 Contribute, individually or as part of a team, to work to identify and solve authentic problems or produce original works using a variety of digital tools and devices.	I	I	R	R	M	• ELA: SL.CC.1 • Mathematics: MP2, MP3, MP4, MP5, MP6, MP7, Literacy Skills for Mathematical Proficiency • Science & Engineering Practices: Asking questions and defining problems; Developing and using models; Analyzing and interpreting data; Using mathematics and computational thinking; Constructing explanations and designing solutions

Analytical and Innovative Thinking (AIT)

Computer Science Standards	Grade Level					Tennessee Academic Standards Connection Examples
	K	1-2	3-4	5-6	7-8	
AIT.1 Identify and define problems and form significant questions for investigation.	I	R	R	M	M	• ELA: RL.KID.1 • Mathematics: MP1, MP2, MP4, MP7, MP8 • Science & Engineering Practice: Asking questions and defining problems • Social Studies: SSP.03
AIT.2 Develop a plan to use technology to find a solution and create projects.	I	I	R	R	M	• ELA: SL.CC.2, W.PDW.6 • Mathematics: MP1, MP2, MP8 • Science & Engineering Practice: Planning and carrying out controlled investigations, constructing explanations and designing solutions
AIT.3 Determine the best technology and appropriate tool to address a variety of tasks and problems.	I	I	R	R	M	• ELA: SL.CC.2, W.PDW.6 • Mathematics: MP5, MP6 • Science & Engineering Practice: Using mathematics and computational thinking
AIT.4 Use multiple processes and diverse perspectives to explore alternative solutions.	I	R	R	M	M	• ELA: SL.CC.2, SL.CC.3, R.RI.CS.6 • Mathematics: MP1,MP4,MP8 • Science & Engineering Practices: Using mathematics and computational thinking; Engaging in argument from evidence; Obtaining, evaluating, and communicating information • Social Studies: SSP.1, SSP.02, SSP.04
AIT.5 Evaluate the accuracy, relevance, appropriateness, and bias of electronic information sources.	I	I	I	R	M	• ELA: SL.CC.2, W.TTP.1, W.TTP.2, W.PDW.6, R.RI.IKI.8 • Mathematics: MP1,MP8 • Science & Engineering Practices: Engaging in argument from evidence;

						Obtaining, evaluating, and communicating information - Social Studies: SSP.02, SSP.03
AIT.6 Collect, organize, analyze, and interpret data to identify solutions and/or make informed decisions.	I	I	R	R	M	- ELA: SL.CC.2, W.TTP.1, W.TTP.2, W.PDW.6 - Mathematics: MP6, MP7, MP8 - Science & Engineering Practices: Analyzing and interpreting data; Constructing explanations and designing solutions - Social Studies: SSP.1, SSP.02, SSP.03, SSP.04
AIT.7 Infer and predict or propose relationships with data.	I	I	R	R	M	- ELA: SL.CC.2, R.RI.IKI.8 - Mathematics: MP1, MP6 - Science & Engineering Practices: Analyzing and interpreting data; Constructing explanations and designing solutions; Engaging in argument from evidence
AIT.8 Identify that various algorithms can achieve the same result and determine the most efficient sequence.	I	I	R	R	M	- Mathematics: MP1, MP2, MP4, MP7, MP8 - Science & Engineering Practice: Using mathematics and computational thinking

Digital Citizenship (DC)

Computer Science Standards	Grade Level					Tennessee Academic Standards Connection Examples
	K	1-2	3-4	5-6	7-8	
DC.1 Advocate, demonstrate and routinely practice safe, legal, and responsible use of information and technology.	I	I	R	M	M	• ELA: W.TTP.1 • Mathematics: MP5 • Science & Engineering Practice:, Obtaining, evaluating, and communicating information
DC.2 Exhibit a positive mindset toward using technology that supports collaboration, learning, and productivity.	I	R	R	M	M	• ELA: SL.CC.1, W.PDW.6 • Science & Engineering Practices: Engaging in argument from evidence; Obtaining, evaluating, and communicating information
DC.3 Exhibit leadership for digital citizenship.	I	R	R	M	M	
DC.4 Recognize and describe the potential risks and dangers associated with various forms of online communications (e.g., cell phones, social media, digital photos).	I	R	R	M	M	• ELA: R.KID.2, R.KID.3, R.RI.IKI.8, W.TTP.2 Mathematics: MP2 • Science & Engineering Practices: Engaging in argument from evidence; Obtaining, evaluating, and communicating information
DC.5 Explain responsible uses of technology and digital information; describe possible consequences of inappropriate use such as copyright infringement and piracy.	I	R	R	M	M	• ELA: R.KID.2, R.KID.3, R.RI.IKI.8, W.TTP.2, W.PDW.6 • Mathematics: MP3 • Science & Engineering Practice: Obtaining, evaluating, and communicating information

Information Storage and Access (ISA)

Computer Science Standards	Grade Level					Tennessee Academic Standards Connection Examples
	K	1-2	3-4	5-6	7-8	
ISA.1 Enter, organize, and synthesize information in a variety of platforms. (e.g., saving, organizing, and storing word documents and spreadsheets)	I	R	R	R	M	• ELA: R.CS.5, R.IKI.9 • Mathematics: MP5, 1.MD.C.5 • Science & Engineering Practices: Developing and using models; Analyzing and interpreting data; Constructing explanations and designing solutions; Obtaining, evaluating, and communicating information • Social Studies: SSP.3
ISA.2 Identify and use a variety of storage media and demonstrate an understanding of the rationale for using a certain medium for a specific purpose.	I	I	R	R	M	• ELA: R.CS.6 • Mathematics: MP5 • Science & Engineering Practice: Using mathematics and computational thinking
ISA.3 Plan and use strategies to access information and guide inquiry.	I	I	R	R	M	• ELA: RL.KID.1 • Mathematics: MP1 • Science & Engineering Practice: Obtaining, evaluating, and communicating information
ISA.4 Locate information from a variety of sources.	I	I	R	R	M	• ELA: R.KID.1, R.IKI.7 • Mathematics: MP5 • Science & Engineering Practice: Obtaining, evaluating, and communicating information • Social Studies: SSP.1
ISA.5 Perform basic searches on databases to locate information.	I	I	I	I	R	• ELA: R.KID.2, R.KID.3 • Mathematics: MP1 • Science & Engineering Practice: Obtaining, evaluating, and communicating information

ISA.6 Select appropriate information sources and digital tools.	I	R	R	M	M	- ELA: R.RI.IKI.8 - Mathematics: MP5 - Science & Engineering Practices: Engaging in Argument from Evidence; Obtaining, evaluating, and communicating information
ISA.7 Use age appropriate technologies to locate, collect, organize content from media collection(s) for specific purposes, such as citing sources.	I	R	R	M	M	- ELA: R.CS.5, R.RI.IKI.8 - Mathematics: MP5 - Science & Engineering Practice: Obtaining, evaluating, and communicating information - Social Studies: SSP.1, SSP.03
ISA.8 Describe the rationale for various security measures when using technology.	I	I	R	M	M	- ELA: R.KID.2, R.RI.IKI.8, W.TTP.2 - Mathematics: MP3

Coding and Computer Programming (CCP) – Grade K	
K.CCP.1	Identify, using appropriate terminology, common physical components of computing systems (hardware). *For example, but not limited to, desktop computers, laptop computers, tablet devices, monitors, keyboards, mice and printers.*
K.CCP.2	Use simple trial and error strategies to identify when a computing device is not working as intended. *For example, but not limited to, if the device does not turn on students can identify if it needs to be charged or is unplugged before saying the device does not work.*
K.CCP.3	Define an algorithm as a list of steps that can be followed to finish a task or solve a problem.
K.CCP.4	Decompose an example problem into smaller sub-problems with teacher guidance or independently.
K.CCP.5	Collaboratively, students can build independence and sophistication using a simple design process *(e.g., Ask, Plan, Do, Reflect)* to illustrate a program's sequence and outcomes.

Coding and Computer Programming (CCP) – Grade 1	
1.CCP.1	Recognize and utilize common physical components of computing systems (hardware) and software concepts using correct terminology. *For example, but not limited to, laptop computers, tablets, monitors, keyboards, printers and software concepts such as, sign-in requirements input output, debug and program.*
1.CCP.2	Use simple trial and error strategies to identify hardware and software problems that occur using appropriate terminology. *For example, but not limited to, an app or program is not working as expected, no sound is coming from a device, or a device will not turn on.*
1.CCP.3	Construct an algorithm by arranging sequential events step-by-step in a logical order.
1.CCP.4	Determine that data *(e.g., numbers, words, colors, and images)* can be stored in computer programs.
1.CCP.5	Collaboratively or individually, students use programming to create simple animated stories or solve pre-existing problems using a precise sequence of instructions and simple loops. *For example, but not limited to, if a dialogue is not sequenced correctly, the animated story will not make sense or using loops in a program to show the life cycle of a butterfly, a loop could be combined with move commands to allow continual but controlled movement of the character.*
1.CCP.6	Decompose larger problems or tasks into smaller sub-problems independently.
1.CCP.7	Collaboratively, students can build independence and sophistication using a simple design process *(e.g., Ask, Plan, Do, Reflect)* to illustrate a program's sequence and outcomes.
1.CCP.8	Compare positive and negative effects computer technology has in the lives of people. Identify ways that programs and/or hardware is used by groups within society. For example, touchscreens are used by children differently than they are used by artists.

Coding and Computer Programming (CCP) – Grade 2	
2.CCP.1	Identify and describe how hardware and software components make up a computing system.
2.CCP.2	Identify, using accurate terminology and debugging strategies, simple hardware and software problems that may occur during use. *For example, but not limited to, if an app or program is not working as expected, no sound or device won't turn on.*
2.CCP.3	Analyze and improve an algorithm that includes sequencing and simple patterns with or without a computing device.
2.CCP.4	Evaluate how computer programs can manipulate stored data *(words, numbers, colors, and images)* with support or independently.
2.CCP.5	Create or revise a computational artifact *(a visualization, a graphic, a video, a program, or an audio recording),* using appropriate attributions for revisions.
2.CCP.6	Define a problem or task, decompose it into smaller sub-problems.
2.CCP.7	Collaboratively, students can build independence and sophistication using a simple design process *(e.g., Ask, Plan, Do, Reflect)* to construct a program's sequence and revise outcomes.
2.CCP.8	Compare positive and negative impacts effects computer technology has in the lives of people.

Coding and Computer Programming (CCP) – Grade 3	
3.CCP.1	Recognize and understand that a series of devices and components form a system of interdependent parts with a common purpose.
3.CCP.2	Describe how and why information is broken up and travels in packets (collections of data).
3.CCP.3	Identify and determine the purpose of a variable and the data that it stores in an algorithm.
3.CCP.4	Using a block of code or script from a previous program, identify the control structures in the algorithm such as loops, and/or conditionals in the code.
3.CCP.5	Using a block of code or script from a previous program, decompose into sections and/or subprograms to make it easier to read or more manageable.
3.CCP.6	Using a block of code or script from a previous lesson, identify sections for the code that may be reused into a new strand of code.
3.CCP.7	Describe ways that programs and/or hardware are used by groups within society. *For example, touchscreens are used by children differently than they are used by artists.*

Coding and Computer Programming (CCP) – Grade 4	
4.CCP.1	Recognize the input and output devices along with the components that form an interdependent system with a common purpose.
4.CCP.2	Demonstrate how information is broken up and can travel in packets through different systems.
4.CCP.3	Using a block of code or script from an existing program, identify the variables in the algorithm to determine if or how these might be manipulated to improve the program.
4.CCP.4	Construct an algorithm to solve a problem that includes control structures such as loops, event handlers, and conditionals collaboratively with or without a computing device.
4.CCP.5	Using a block of code or script that has been used in a previous program or algorithm, identify sections that can be reused into a new block or script of code.
4.CCP.6	Use existing code and identify sections of code that can be used to remix into a new program with proper attributions for efficiency.
4.CCP.7	Describe ways that hardware and software are used by various members of society including accessibility features. For example, voice commands can be used for accessibility or convenience.

Coding and Computer Programming (CCP) – Grade 5	
5.CCP.1	Identify and describe the role of various input and output devices and components that are within an interdependent system with a common purpose.
5.CCP.2	Investigate and trace a bundle of information through a series of packets and different systems via a protocol.
5.CCP.3	Decompose (break down) complex real-world problems in multiple ways that use variables to develop a solution or procedure based on data.
5.CCP.4	Create an algorithm which includes control structures to solve a problem using visual block-based and/or text-based programming language both collaboratively and individually.
5.CCP.5	Decompose complex code into subsections or subprograms for reuse into other programs.
5.CCP.6	Decompose a piece of code with the intent to debug a section of code.
5.CCP.7	Formulate alternative uses for software and hardware for various members of society.

Coding and Computer Programming (CCP) – Grade 6-8	
CCP.1	Identify the advantages, disadvantages and unintended consequences of computing devices.
CCP.2	Analyze the relationship between human and computer interactions to improve the device. *For example, student A watches student B use a simple communication device. Student A updates the tool for improved use.*
CCP.3	Identify and describe multiple considerations and tradeoffs when designing or selecting computing system, such as functionality, cost, size, speed, accessibility, and aesthetics.
CCP.4	Construct optimized models of computing systems.
CCP.5	Create structured processes to troubleshoot problems with computing systems.
CCP.6	Define protocols in relation to a set of rules.
CCP.7	Construct protocols that can be used to share information between people or devices. *For example, a binary communication protocol using lights.*
CCP.8	Compare the relative strengths and weaknesses of unique protocols considering security, speed, and reliability.
CCP.9	Create models of networks that include packets and domain name server (DNS).

CCP.10	Identify steps to ensure security measures are in place to safeguard online information.
CCP.11	Create cyphers to encrypt data that can be transferred between users.
CCP.12	Explain how encryption can be used to safeguard data that is sent across a network.
CCP.13	Evaluate the accuracy and precision of various forms of data collection.
CCP.14	Identify and define the limiting factors to specific forms of data collection.
CCP.15	Describe how different formats of stored data represent tradeoffs between quality and size.
CCP.16	Represent data using different encoding schemes, such as binary, Unicode, Morse code, shorthand, student-created codes.
CCP.17	Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools.
CCP.18	Revise variables and constants in computational models to more accurately reflect real-world systems. For example in an ecosystem model, introducing predators as a new variable.

CCP.19	Solicit and integrate peer feedback as appropriate to develop or refine a program.
CCP.20	Compare different algorithms that may be used to solve the same problem in terms of their speed, clarity, and size.
CCP.21	Provide proper attribution when code is borrowed or built upon.
CCP.22	Interpret the flow of execution of algorithms and predict their outcomes.
CCP.23	Design, develop, and present computational artifacts such as mobile applications that address social problems both independently and collaboratively.
CCP.24	Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches. *(Clarification: At this level, students may use block- based and/or text-based programming languages.)*
CCP.25	Identify the purpose of variables in relation to programming
CCP.26	Create variables that represent different types of data and manipulate their values.
CCP.27	Define and use procedures that hide the complexity of a task and can be reused to solve similar tasks. *(Clarification: Students use and modify, but do not necessarily create, procedures with parameters.)*

CCP.28	Decompose a problem into parts and create solutions for each part.
CCP.29	Use an iterative design process *(e.g., define the problem, generate ideas, build, test, and improve solutions)* to solve problems, both independently and collaboratively.
CCP.30	Analyze the positive and negative impacts of computing technology.
CCP.31	Recognize there are tradeoffs in computing.
CCP.32	Explain how social interactions can allow for multiple viewpoints.
CCP.33	Demonstrate an understanding of digital security.

Secondary
Computer Science Standards

Introductory Computer Science Standards

1) Accurately read, interpret, and demonstrate adherence to safety rules, including (1) rules published by the National Science Teachers Association (NSTA), (2) rules pertaining to electrical safety, (3) Internet safety, (4) Occupational Safety and Health Administration (OSHA) guidelines, and (5) state and national code requirements. Be able to distinguish between rules and explain why certain rules apply.

2) Identify and explain the intended use of safety equipment available in the classroom. For example, demonstrate how to properly inspect, use, and maintain safe operating procedures with tools and equipment.

3) Demonstrate understanding of electrical circuits and devices, and relate to the physical laws (such as Ohm's Law and power laws) that govern behaviors of electrical circuits and devices. Accurately apply these physical laws to solve problems. For example, calculate the resistance of a DC circuit with a given DC voltage and current.

4) Assemble the required connections of electronic test equipment to properly test the operation of basic electronic circuit behavior and performance, using equipment such as a digital multimeter. For example, demonstrate the proper use of a digital multimeter by measuring resistance of a circuit in a typical computer system; compare this finding by calculating the resistance given the voltage and current.

5) Distinguish between the binary and hexadecimal counting systems. Using appropriate units, provide examples of each system and identify specific instances when IT professionals rely on them.

6) Explain the functions of gates in logic circuits (e.g., AND, OR, NOT). For example, construct a truth table for the seatbelt warning light in an automobile.

7) Research various occupations in information technology industries, such as programmers, web designers, webmasters, networking administrators, computer systems administrators, telecommunications line installers, and informational security analysts. Compose an informative table or chart that includes the following: work activities typically performed, tools and technology used, nature of work environment, and the knowledge and skills needed for success.

8) Explore various professional societies related to information technology and identify the services and benefits provided by each member. Create a table that lists their purposes, benefits to membership, and any certifications affiliated with the organization. For example, investigate the Institute for Electrical and Electronics Engineers (IEEE), Computing Technology Industry Association (CompTIA), and the Association for Computing Machinery (ACM).

9) Drawing on multiple sources (i.e., internet, textbooks, videos, and journals), research the history of the Internet. Create a timeline or infographic, illustrating the Internet's historical evolution from its inception to the present time. Discuss the needs that led to the creation of the Internet; discuss both the benefits and disadvantages of the Internet to society, as well as potential implications for the future. Provide examples drawn from the research to support claims.

10) Drawing on multiple sources (i.e., internet, textbooks, videos, and journals), research the history and development of operating systems (e.g., Microsoft Windows, Linux, UNIX). Create a presentation, illustrating their historical evolution, from their inceptions to the present, citing information found in research. Compare and contrast the general capabilities of a variety of operating systems, and explain how their designs and functionalities have improved over time.

11) Demonstrate an understanding of basic web terminology and concepts. Practice explaining these terminologies and concepts by creating methods to help students learn and remember the information. For example, students should be able to explain the purpose of terminology such as server, domain name system (DNS) , internet service provider (ISP), hardware and software connective devices, cloud computing, remote access protocols, map protocols, content management systems (CMS), cascading style sheets (CSS), and social networking terms.

12) Demonstrate a basic understanding of computer hardware components. Identify these components using pictures or actual models and briefly explain the function of each. Components should include, but are not limited to:
    a. Hardware used for input and output
    b. Hardware inside the computer case
    c. Motherboard
    d. Processor and the chipset
    e. Storage devices (e.g., primary, secondary)
    f. Expansion cards
    g. Electrical system

13) Demonstrate a basic understanding of computer networking. For example, explain the types of networks and what a client-server environment is.

14) Identify, explain, and demonstrate the use of common keyboard shortcuts. Create a quick reference guide that would be user-friendly for a novice web designer. For example, students may create a multiple column table showing keyboard shortcuts for navigation, text editing, and text formatting. The table would identify which shortcuts are applicable to using Windows versus Mac OS.

15) Synthesize research of historical and significant milestones that influenced the evolution of cloud computing. Create an annotated timeline or visual graphic illustrating significant time periods and major impacts of technology trends that influenced the development of cloud computing. Use academic research and news media citing specific textual evidence from research.

16) Identify, describe, and effectively summarize cloud technology roles including: cloud computing customer, cloud service provider and cloud service partner. Create a written report or visual depiction outlining the characteristics of each.

17) Research the features and requirements of the four main deployment models for cloud technology: public, private, community, and hybrid. Create a graphic illustration showing the roles of each and describe their differences.

18) Consult a variety of sources to describe how virtualization, storage, networking and databases in cloud technologies are used. Sources may include textbooks, manuals, websites, video tutorials, and more. Create a visual display with accompanying text comparing these methods.

19) Explore the onset of the Internet of Things (IoT) and explain how it is enabled by sensors, actuators, communication devices and computers that exchange and process data and can interface with users in a most instinctual way. Using a specific example, summarize in a graphic illustration or narrative how the IoT combines information, automation, computation, software, sensing, and networking to make traditional processes more efficient.

20) Consult internet forums, textbooks, industry journals and other instructional materials to research the importance of developing and implementing databases, data collection systems, data analytics and other strategies that optimize statistical efficiency and quality. Write a brief paper that discusses the importance of these services in business today. Provide specific examples to support the claims.

21) There are different versions of the web design and development process. For example, most versions of the web design and development process involve project definition, site structure, visual design, site development, testing, refining, and launch. Using various resources, research, identify, and explain the steps involved in the process. As a class, develop an agreed-upon framework for applying the logical thought process to web design projects in the form of a flowchart or logic model, justifying the reasoning behind each step. Explain why it is an iterative process and always involves refinement.

22) Research, identify, and describe the specific activities involved at each step of the troubleshooting process, including by not limited to:1) gather information from the user or operator and back up data, 2) verify the problem exists, 3) isolate the cause of the problem and generate alternative solutions, 4) plan a solution and resolve the problem, 5) verify that the problem was resolved and prevent a future occurrence, and 6) document findings, resolution, and preventative maintenance plan. Explain why it is important to document the process throughout.

23) Demonstrate an understanding of flowcharts and know what various symbols mean. Identify a problem that a programmer would solve using the logical thinking process, and create a flowchart that would guide the code development. For example, create a flowchart that incorporates at least three decisions, or paths, to solve a problem.

24) Explore how teams are formed to complete and manage web design and development projects. Using the information gained from research, identify and explain various roles and responsibilities for members of a web design and development team. Include why teams are more efficient than individuals in the web design and development process. Present the findings to classmates.

25) Synthesize common principles and templates for successful project management. Explain, using examples, why strong management skills are important in the web design and development process.

26) Research and identify the skills that are required to communicate effectively with a client. Develop a questionnaire that would be used to determine the needs of a client for a prospective web development project. Using the questionnaire, conduct mock client interviews with classmates and provide each other with constructive feedback to revise the questionnaire and process.

27) As a team, list primary rules to guide writing content that is appropriate for a web site publication. Apply these rules to a variety of web-based writing assignments throughout the course. For example, develop and maintain a blog throughout the course to practice appropriate writing techniques and style for web publication.

28) Given a specific client's vision, create a simple web site using a content management system (CMS) such as WordPress. Follow the multistep process to download the software application of choice, and demonstrate how to upload and store files. Practice proofreading and critiquing other classmates' sites, and provide constructive feedback on one another's writing and layout design.

29) Drawing on multiple sources (i.e., internet, textbooks, videos, and journals), research the various social, legal, and ethical issues encountered by IT professionals. Using these findings, identify the roles and responsibilities one must consider while developing a prospective project or addressing an IT problem. For example, web developers and programmers must apply copyright laws and understand uses of open source software.

30) Using various sources (i.e., internet, textbooks, videos, and journals), research and identify reasons as to why data security should be a priority to technology professionals through demonstrating an understanding of information security fundamentals on Confidentiality, Availability, and Integrity.

31) Demonstrate an understanding of the various security breaches that can occur with the Internet. Prepare a text explaining enterprise-level security, the purpose of encryption, and the protocols that can be implemented to secure web sites. Evaluate personal privacy issues versus employers' rights to regulate computing resources.

32) Identify various security practices for computer and network systems, such as how to control access to secured resources and computer resources. Give specific examples of methods that an administrator can use, like encryption techniques, basic input/output system (BIOS) features, and strategies for dealing with malware.

33) Understand and demonstrate the effective use of file and folder management techniques to maintain directory structure for a web site. Describe the most efficient methods for digital file management, including the use of site root and subfolders for assets (e.g., images, templates, CSS).

34) Explore and identify various languages, such as Python, HTML, PHP, C++, Visual Basic, Java, JavaScript, and C #. Explain how programmers use these languages to solve a variety of IT problems, furnishing examples of how they are applied.

35) Using various resources, research, identify, and explain the steps involved in the software development life cycle, including but not limited to: planning, designing, coding, testing, deployment, and maintenance. Explain why it is an iterative process and always involves refinement.

36) Demonstrate an understanding of how batch files function within a programming environment. Identify common commands to create code for batch files (e.g., title, echo, echo off, pause, CLS, ipconfig, and ping). For example, list various scenarios for using batch files to complete specific programming tasks. Create and execute batch file code to perform one of the tasks identified.

Beginning Computer Science Standards

37) Using news articles and instructional materials, investigate key milestones in the development of computers and logical devises. Create and present a document and/or illustration depicting the timeline of development that led to modern-day operating systems, programmable controllers, and widespread digital communications via the Internet and wireless networks, citing specific textual evidence.

38) Compare and contrast the benefits, features, and typical applications of common modern programming languages and environments. Craft an argument to defend the choice of a certain language to solve a particular problem, developing claim(s) and counterclaim(s) with specific textual evidence and reasoning.

39) Using news articles and text of legislation, analyze ethical programming practices, including but not limited to the issues of confidentiality, privacy, piracy, fraud and misuse, liability, copyright, open source software, trade secrets, and sabotage. For example, research and report on the effects of unethical programming practices on a business.

40) Differentiate between system-level and application solutions, and identify an appropriate code-based strategy to solve a given problem. For example, given a file management problem, determine when a command-line script will be more efficient than a high-level program solution.

41) Apply the system management tools present in a programming development environment to:
    a. Select the most appropriate programming language for the task at hand
    b. Develop syntactically correct program code using current best practices and emerging classes of development techniques
    c. Use a compiler to interpret the source code and produce executable program code

42) In the process of developing and implementing programming solutions, develop strategies that work within the constraints of major operating system fundamentals, such as:
    a. Security protocols and procedures for accessing files and folders
    b. File management syntax requirements, including but not limited to creating, naming, organizing, copying, moving, and deleting files
    c. File naming conventions, as they apply across multiple software applications and file types.

43) Write pseudocode and construct a flowchart for a process before starting to develop the program code. For example, code and flowchart a simple process that takes an integer and report whether it is odd or even.

44) Organize and develop a plan to acquire and manage the data values for a process, including the following:
    a. Data types, such as string, numeric, character, integer, and date
    b. Program variable names
    c. Variables and constants
    d. Arrays (at least one- and two-dimensional), subscripts
    e. Input from files and user responses
    f. Output to files and reports

45) Using a programming language specified by the instructor, convert the pseudocode for a selected process to program code, incorporating at least three of the following structures, the need for which will be dictated by the assigned problem(s) and process(es). The resulting code design can be event-driven, object-oriented, or procedural.
    a. Operations and functions (user-defined and/or library)
    b. Repetition (loops)
    c. Decision (if…else, case)
    d. Recursion

46) Verify the correct operation of the resulting program code with several test cases:
    a. All valid values
    b. Error trapping of invalid values
    c. Error trapping of invalid program operation
    d. Troubleshooting/remedying program problems

47) Compile the necessary documentation to understand the nature of a computer programming problem and the customer/client specifications for the request and summarize in an informational text. This will include evidence of the scope of the problem, its attendant input and output information, the required system processing, and the software specifications involved.

48) Analyze a given problem and develop a coherent strategy in the form of a project plan to meet the customer/client's need. The plan will include, but will not be limited to, defining the project scope as addressed by the problem documentation, identifying software development and implementation issues, timeline and benchmarks for design, and addressing issues associated with software maintenance and life cycle.

49) In the software development process, articulate the nature of the program designs by creating documentation that addresses topics including but not limited to:
    a. The procedural, object-oriented, event-driven, or other nature of the various portions of the resulting application
    b. The data structures used for inputs, outputs, and internal manipulations
    c. The algorithms and guiding formulas used
    d. Constraints on accurate operation and results
    e. Modular designs that enable portability
    f. Interface details that permit ready maintenance and upkeep

50) Apply principles of quality assurance during application development to certify bug tracking, audit trails, testing results, and other quality considerations. Annotate each quality assurance task with evidence from best practices endorsed by industry or research.

51) Document the security risks associated with new applications and evaluate the severity of the risk involved in each, including but not limited to:
    a. Identifying threats to information systems facilities, data communications systems, and other applications
    b. Adhering to federal and state legislation pertaining to computer crime, fraud, and abuse
    c. Providing means for preserving confidentiality and encryption of sensitive data
    d. Detailing steps to recover from routine errors or catastrophic failures, such as might be caused by a malicious computer virus.

Intermediate Computer Science Standards

52) Evaluate at least two software development environments (SDEs) that are tailored to different programming languages on the basis of their suitability for a range of programming tasks, ease of use, and how ubiquitous they are within the IT community. Document in an oral presentation the similarities and differences between the two, and the features that lend themselves to the chosen programming languages. For example, students assigned to code a basic database interface can compare the benefits and features of a freeware SDE such as JDeveloper and a commercial SDE like Microsoft Visual Studio.

53) Investigate the typical process around creating new software within a software development

environment. Describe and furnish examples of the steps taken within the SDE to guarantee reliable output, from prototyping and authoring to deployment and debugging.

54) Administer the process of creating new software within a software development environment to manage the prototyping, authoring, revising, compiling, testing, deploying, and debugging of student-developed software. For example, for an object-oriented payroll program assignment (retrieving file data to produce a run of paychecks and paystubs for a small business), perform and document the steps taken within the SDE to ensure the reliable and accurate output of paychecks.

55) Synthesize information from a range of sources (including original tests and simulations) to critique the features of different software development life cycles (agile, iterative, and sequential types). Using domain-specific terminology, explain to a technical audience the distinguishing features of each that make one more appropriate for certain types of applications.

56) For a selected assignment or project involving the development of original software, choose and defend a strategy to follow for the program's development life cycle. At the completion of the assignment, offer recommendations for other environments and alternative strategies that could improve the development process.

57) Research common and best-practice techniques in programming analysis, design, and implementation. Drawing on model practices used by businesses and industry, employ analysis, design, and implementation techniques to satisfy a programming need, using an appropriate software lifecycle model.

58) Employ a requirement management tool during a program's development life cycle, documenting the evolving versions, storage attributes, system elements, status tracking, and access permissions afforded by the tool, as well as the successful attainment of the project vision.

59) For a given programming assignment, choose and defend a programming language with regard to the language's capabilities and suitability to task, availability, portability, maintainability, and cost.

60) For the assignment outlined in standard 8, identify the method of data processing most appropriate for the task (e.g., batch, interactive, or event-driven). For example, a weekly payroll application would handle its data differently (i.e., batch processing) than a web-based search engine (i.e., interactive processing), and still differently than a microprocessor control program for a washing machine (i.e., event driven).

61) Define the specifications of the data management plan, including variables (naming, scope, and types), validation measures (to protect the data from corruption), and data handling (storing, input/output, and back-up). For example, programs handling historical temperature data would be best suited to floating point values stored in multidimensional arrays, written to permanent storage, and displayed with limited precision.

62) For a selected programming assignment involving an object-oriented language, design and define the classes, objects, properties, methods, and inheritance structures prior to the start of the development cycle. Revise the plan (modifications, additions, and subtractions) as needed throughout the development cycle.

63) For selected programming assignments, create, edit, and improve documentation for technical support intended for fellow programmers, including within the program code itself as well as within supplemental documents. For example, for a lawn sprinkler system microcontroller, the technical documentation would define the variables, functions and subroutines, and the critical events.

64) For selected programming assignments, create, edit, and improve end-user documentation. End- user documentation would include how to interact with the user interface, the capabilities and limitations of the system, and the required conditions for successful operation.

65) Incorporate structured, object-oriented, and event-driven programming techniques that employ sequence, selection, and/or repetition (loops) to solve programming projects.

66) For each programming task, consider and defend the choice of various programming approaches (such as data-driven or event-driven, top-down or bottom-up), citing examples from the syntax illustrating the chosen approach.

67) Design and develop an app for a mobile computing device, using an online programming interface, such as AppMakr, BuzzTouch, Appsbar, PhoneGap, or AppYet.

68) During the development, testing, and deployment of a new program, implement checks for data and procedure accuracy, correctness, currency, and relevance, making and documenting revisions where justified.

69) Analyze the code written by another programmer to create a flowchart, suggesting points of confusion or generality in the program that could become problematic in future revisions. Cite specific examples in the code to support recommendations.

70) Conduct quality testing of program code, striving for satisfactory results at four levels or perspectives:

   a) Unit (component/module level verifications)
   b) Integration (verifying the interfaces between components, adding one at a time)
   c) System (verifying that the whole package meets the requirements and specifications without corrupting other systems)
   d) Acceptance (customer satisfaction)

71) Design, manage, and develop a course-long programming project pre-approved by the instructor. The project will embody a variety of strategies and resources taught in this course, and require periodic reviews, status reports, and final project presentation. Use a software development environment to manage, document, test, deploy, and maintain the resources and assets of the finished project.

Advanced Computer Science Standards

72) A student will have a Personalized Learning Plan that identifies their long-term goals, demonstrates how the Work-Based Learning (WBL) experience aligns with their elective focus and/or high school plan of study, addresses how the student plans to meet and demonstrate the course standards, and addresses employability skill attainment in the following areas:
    a. Application of academic and technical knowledge and skills (embedded in course standards)
    b. Career knowledge and navigation skills
    c. 21st Century learning and innovation skills
    d. Personal and social skills

73) Research a company or organization that employs computer programmers or specializes in software design and development solutions. Companies could range from large software developers, to niche organizations that retain programmers on staff to serve their particular clients' needs. For the chosen company, cite specific textual evidence from the company's literature, as well as available press coverage (if available) to summarize:
    a. The mission and history of the organization
    b. Headquarters and organizational structure
    c. Products or services provided
    d. Credentials required for employment and how they are obtained and maintained
    e. Policies and procedures
    f. Reports, newsletters, and other documents published by the organization
    g. Website and contact information

74) Analyze the requirements and qualifications for various programming and development job postings identified from specific company websites or online metasearch engines. Gather information from multiple sources, such as sample resumes, interviews with professionals, and job boards, to determine effective strategies for realizing career goals. Create a personal resume modeled after elements based on the findings above, then complete an authentic job application as part of a career search or work-based learning experience.

75) Participate in a mock interview. Prior to the interview, research tips on dress and grooming, most commonly asked interview questions, appropriate conduct during an interview, and recommended follow-up procedures. Upon completion of the interview, write a thank you letter to the interviewer in a written or email format.

76) Investigate current issues surrounding the use of software applications to collect and track user data. Explore a range of arguments concerning privacy rights as they relate to the mining of personal data; determine when it is ethical and legal to collect data for profit versus for security purposes. Advance an original argument that debates the pros and cons and summarizes the potential ramifications for clients, users, the public, and one's own personal reputation, drawing on evidence gathered from news media, company policies, and state and federal laws.

77) Research a case study involving an ethical issue related to intellectual property rights. Examine a variety of perspectives surrounding the issue, then develop an original analysis explaining the impact of the issue on those involved, using persuasive language and citing evidence from the research. Potential issues include copyright infringement, piracy, plagiarism, art licensing, creative commons, and the state/federal laws that govern them.

78) In teams or individually, develop a written proposal for an original program or software application that involves advanced refinement and transfer of skills and knowledge acquired in previous *Programming & Software Development* courses. The proposal should be narrative in nature but supplemented by relevant data and graphic illustrations as needed, such as flowcharts of development processes and diagrams or sketches of what the end product would resemble. Sample projects include: developing a mobile app; designing an animation package or plug-in; writing an original game program; or any other programming-based project. Present the proposal to the class, and continually revise based on feedback from peers.

79) Throughout the design and development process, develop supplementary documents, presentations, and strategies to support the production and promotion of the program, app, or product. Identify the target market for the product, and devise a tentative plan to inform, promote, and convince prospective users of the product's functions and value. Research marketing plan templates and sample presentations, and synthesize information to produce an original plan outlining how the team intends to market the product once it is finished.

80) Apply coding skills learned in previous courses to novel contexts and development environments. For example, develop skills in an emerging technology that would support the completion of the course project, or learn a new programming language not previously studied in order to enhance the functionality of the product.

81) In the course of developing the project, regularly test for functionality, compatibility, and other design aspects related to user friendliness. Conduct and document the proper code validation to resolve errors encountered in the design process.

82) Analyze the code written by another team member or peer and create a flowchart for suggesting changes to improve functionality. Cite specific examples in the code to support recommendations.

83) Research and test for potential security threats related to the intended uses of the app, program, or product. For example, if a mobile app is developed, determine the most common security threats and identify areas of vulnerability in the product that could be remedied by adjusting for the proper code, patching, or system update. If possible, develop and incorporate security measures into the final product to ensure user safety.

84) Create a portfolio, or similar collection of work, that illustrates mastery of skills and knowledge outlined in the previous courses and applied in the practicum. The portfolio should reflect thoughtful assessment and evaluation of the progression of work involving the application of steps of the design process, as outlined by the instructor. The following documents will reside in the student's portfolio:

    h. Personal code of ethics
    i. Career and professional development plan
    j. Resume
    k. Project proposal with supporting documents
    l. List of responsibilities undertaken through the course
    m. Examples of visual materials developed and used during the course (such as drawings, models, presentation slides, videos, and demonstrations)
    n. Marketing plan
    o. Description of technology used, with examples if appropriate
    p. Periodic journal entries reflecting on tasks and activities
    q. Feedback from instructor and/or supervisor based on observations

85) Produce technical reports highlighting the purpose, content, and use of the app, program, and product developed for this course. Cite evidence from multiple authoritative sources in order to justify design and development decisions and maximize the user experience. Incorporate supporting graphics, sketches, and data as needed to summarize the technical specifications of the product.

86) Upon completion of the practicum, develop a technology-enhanced presentation showcasing highlights, challenges, and lessons learned from the experience. The presentation should be delivered orally, but supported by relevant graphic illustrations, such as diagrams, flowcharts, and/or market data on the target users. Prepare the presentation in a format that could be presented to both a technical and a non-technical audience, as well as for a career and technical student organization (CTSO) competitive event.