FUNDAMENTALS OF WEBPAGE DESIGN AND DEVELOMPENT / ADVANCED WEBPAGE DESIGN AND DEVELOPMENT

(Course Codes: 5031, 5032)

As aligned to:

South Carolina Computer Science High School Process and Content Standards

The South Carolina Computer Science and Digital Literacy Process Standards should be integrated into every grade level within the South Carolina Computer Science and Digital Literacy Content Standards. Because the Process Standards drive the pedagogical component of teaching and serve as the means by which students should demonstrate understanding of the content standards, the process standards must be incorporated as an integral part of overall student expectations when assessing content understanding.

A computer science literate student can:

1. Foster an inclusive computing culture.
   a. Recognize that equitable access to computing benefits society as a whole.
   b. Consider others' perspectives as well as one's own perspective when developing computational solutions.
   c. Consider the needs of a variety of end users regarding accessibility and usability.

2. Collaborate around computing.
   a. Select appropriate technological tools that can be used to collaborate on a project.
   b. Collaborate productively with individuals of varying perspectives, skills, and backgrounds.
   c. Set and implement equitable expectations and workloads when working in teams.
   d. Integrate constructive feedback while working in teams.

3. Recognize, define, and analyze computational problems.
   a. Recognize when it is appropriate to solve a problem computationally.
   b. Make sense of computational problems and persevere in solving them.
   c. Relate computational problems to prior knowledge.
   d. Recognize that there may be multiple approaches to solving a problem.
   e. Approach problem solving iteratively, using a cyclical process.

4. Create, test, and refine computational artifacts.
   a. Consider the purpose of computational artifacts for practical use, personal expression, and/or societal impact.
   b. Recognize when to use the same solution for multiple problems.
   c. Test computational artifacts systematically by considering multiple scenarios and using test cases.

d. Approach troubleshooting systematically.
e. Consider performance, reliability, usability, and accessibility when evaluating and refining computational artifacts.

5. Communicate about computing.
   a. Select and use appropriate technological tools to convey solutions to computing problems.
   b. Communicate about computational processes and solutions using appropriate terminology consistent with the intended audience and purpose.
   c. Articulate ideas responsibly by observing intellectual property rights and giving appropriate attribution.

South Carolina Computer Science High School Content Standards

Computing Systems

Standard 1: Examine how hardware and software contribute to computing devices solving relevant problems.
HS3.CS.1.1
Recommend modifications for existing computing devices and software to improve functionality for end users.
HS4.CS.1.1
Develop a solution to a given problem using appropriate hardware and software (e.g., sensor devices, Wi-Fi capabilities, specialized displays, runtime modules, operating systems, application programming interfaces (APIs)).

Networks and the Internet

Standard 1: Evaluate data transmission across networks, including the Internet.
HS1.NI.1.1
Describe how hardware, software, and protocols work together for transmitting data across networks.
HS2.NI.1.1
Compare and contrast common network protocols (e.g., Internet Protocol (IP), File Transfer Protocol (FTP), Transmission Control Protocol (TCP), Domain Name System (DNS), Hypertext Transfer Protocol (HTTP), Hypertext Transfer Protocol with Secure Sockets Layer (HTTPS), Simple Mail Transfer Protocol (SMTP), Post Office Protocol (POP), Internet Message Access Protocol (IMAP), Telnet, Secure Shell Protocol (SSH)) (Arkansas, 2016).

Standard 2: Evaluate cybersecurity threats and appropriate security measures across networks.
HS3.NI.2.1
Research security measures (i.e., hardware, software, and practices that control access to data and systems) to combat a variety of cybersecurity vulnerabilities (CSTA, 2017).
HS3.NI.2.2
Evaluate various security measures, considering tradeoffs between the usability and security of a computing system. (e.g., a web filter that prevents access to many educational sites but keeps a campus' network safe) (CSTA, 2017).

Data and Analysis - none

Algorithms and Programming

Standard 1: Design algorithms that can be adapted to express an idea or solve a problem.
HS1.AP.1.1
Create flowcharts and/or pseudocode to express a problem or idea as an algorithm.
HS3.AP.1.1
Adapt predefined algorithms to solve computational problems.
HS4.AP.1.1
Evaluate algorithms in terms of efficiency, correctness, and clarity (CSTA, 2017).

Standard 2: Build a combination of control structures that supports complex execution, readability, and program performance.
HS1.AP.2.1
Trace the flow of execution of a program that uses a combination of control structures (e.g., conditionals, loops, event handlers, recursion).
HS2.AP.2.1
Design and iteratively develop programs that combine control structures (e.g., conditionals, loops, event handlers, recursion).
HS3.AP.2.1
Justify the selection of specific control structures explaining the benefits and drawbacks of choices made (e.g., tradeoffs involving implementation, readability, and program performance).

HS1.AP.2.2
Trace the flow of execution of a program that uses a variety of programming constructs (e.g., procedures, modules, objects).
HS2.AP.2.2
Design a solution through systematic analysis using programming constructs (e.g., procedures, modules, objects).
HS3.AP.2.2
Justify the selection of specific programming constructs, explaining the benefits and drawbacks of choices made on the program's execution.

Standard 3: Divide a task into sets of functional units that can be reused to compose a complex solution.
HS1.AP.3.1
Decompose tasks into smaller, reusable parts to facilitate the design, implementation, and review of programs.
HS2.AP.3.1
Develop code to solve the smaller parts of a decomposed task that can be reused to solve similar problems (e.g., procedures, functions, objects).

HS3.AP.3.1
Build a complex solution to a problem that incorporates reusable code (e.g., student created, application programming interfaces (APIs), libraries).

Standard 4: Plan, build, test, refine, and document programs using text-based coding languages to solve problems with varying degrees of difficulty.
HS1.AP.4.1
Plan and develop programs for a variety of audiences using a process that incorporates development, feedback, and revision.
HS2.AP.4.1
Plan and develop a program that addresses potential security issues.
HS3.AP.4.1
Plan and develop a program that is accessible across multiple computing platforms (e.g., iOS, Unix, Windows, web-based).

HS1.AP.4.2
Seek and incorporate feedback to refine a solution (e.g., users, team members, code review, teachers).
HS3.AP.4.2
Evaluate and refine programs to make them more usable, functional, and accessible.

HS1.AP.4.3
Recognize the variety of documentation methods available while developing a program (e.g., inline comments, procedure header, purposeful naming).
HS2.AP.4.3
Document programs in order to make them easier to follow, test, and debug.
HS3.AP.4.3
Document programs that use non-user-created resources (e.g., code, media, libraries) giving attribution to the original creator.

HS1.AP.4.4
Examine licenses (i.e., permissions) that limit or restrict use of resources (e.g., freeware, shareware, open source, creative commons).
HS2.AP.4.4
Discuss the implications of using licensed resources in a developed solution.

Standard 5: Choose data types and data structures based on functionality, storage, and performance tradeoffs.
HS1.AP.5.1
Justify and use appropriate data types (i.e., primitive and non-primitive) in simple programs.

Impact of Computing

Standard 2: Evaluate the evolving legal and ethical tradeoffs that shape computing practices.

HS1.IC.2.1

Select the most appropriate means of communication for given situations (e.g., personal versus professional communication, communication with teachers and employers).

Source: South Carolina Computer Science High School Content Standards as found at https://ed.sc.gov/scdoe/assets/File/instruction/standards/Computer%20Science/SBE_FINAL_DRAFT_South_Carolina_Computer_Science_Standards_for_High_School_August_2018.pdf.