STRANDS AND STANDARDS

COMPUTER SCIENCE PRINCIPLES

Course Description

Computer Science Principles introduces students to the breadth of the field of computer science. In this course, students will learn to design and evaluate solutions and to apply computer science to solve problems through the development of algorithms and programs. They will use data to discover new knowledge. Students will also explain how computing innovations and computing systems, including the Internet, work, explore their potential impacts, and contribute to a computing culture that is collaborative and ethical.

Intended Grade Level	9 - 12
Units of Credit	0.5
Core Code	35.02.00.00.035
Concurrent Enrollment Core Code	35.02.00.13.035
Suggested Prerequisite	Digital Literacy
Skill Certification Test Number	803
Test Weight	0.5
License Area of Concentration	CTE and/or Secondary Education 6-12
Required Endorsement(s)	
Endorsement 1 or	Information Technology Systems
Endorsement 2 or	Intro to Computer Science
Endorsement 3 or	Programming & Software Development
Endorsement 4	Web Development

*These Strands & Standards are framed after the Fall 2020 AP Computer Science Principles Course. The AP CSP Exam Descriptions can be found HERE.

ADA Compliant: September 2023

STRAND 1

Creative Development

Collaboration is crucial when developing computing innovations, because having multiple perspectives offers additional opportunities to find solutions.

Standard 1

Collaboration

- Explain how collaboration affects the development of a solution.
- Collaborate in the development of solutions.

Standard 2

Program Function and Purpose

- Investigate the situation, context, or task.
    - Investigate the purpose of a program.
    - Understand how to break down program specifications into smaller tasks using top-down design and pseudocode.
- Generalize data sources through variables.
    - Understand the uses of different data types (examples: integer, float/double, characters/strings, boolean, etc.)
- Explain how a code segment or program functions.

Standard 3

Identifying and Correcting Errors

- Identify and correct errors in algorithms and programs, including error discovery through testing.
- Identify different types of errors such as logic, run-time, and syntax errors.

Performance Skill

Students will use problem solving skills and collaboration to explore the purpose of a program and identify and correct errors in a given program.

STRAND 2

Computing and Data

Processing data is the main benefit of computer use.

Standard 1

Hardware/Software

- Explain the differences between hardware and software and how they relate to input, storage, processing, and output.
- Understand the different file sizes (bit, byte, kilobyte, megabyte, gigabyte, terabyte, and petabyte).

Standard 2

Binary Numbers

- Calculate the binary (base 2) equivalent of a positive integer (base 10) and vice versa.
- Compare and order binary numbers.

Standard 3

Data Compression

- Lossy - Compression algorithm in which some of the data from the original file is lost.
- Lossless - Compression algorithm in which file size is reduced without any quality loss.

Performance Skill

Students will differentiate between hardware and software, convert decimal to binary and binary to decimal, and identify when to use lossy vs. lossless compression.

STRAND 3

Algorithms and Programming
Algorithms and programming languages are essential for solving problems and completing tasks.

Standard 1

Variables and Assignments

- Use variables of different data types (examples: integer, float/double, characters/strings, boolean, etc.)
- Convert data types to other data types.
- Determine the value of a variable as a result of an assignment.

Standard 2

Mathematical Expressions

- Implement arithmetic operators (=, +, -, *, /, and MOD) and order of operations (PEMDAS).

Standard 3

Input / Output

- Receive and store user input.
- Print to console

Standard 4

Strings

- Evaluate expressions that manipulate strings.
  - String concatenation joins together two or more strings end-to-end to make a new string.

Standard 5

Boolean Expressions

- Write and evaluate expressions using relational operators (==, ≠, >, <, ≥, and ≤).
- Write and evaluate expressions using logical operators (AND, OR, NOT).

Standard 6

Conditionals

- Write conditional statements, such as IF statements and ELSE IF statements.
- Determine the result of conditional statements.

Standard 7

Iteration/Looping

- Write iteration statements, such as for loops and while loops.
- Determine the result of iteration statements.

Standard 8

Calling and Developing Procedures/Functions/Methods

- Write statements to call Procedures/Functions/Methods
- Determine the result of a Procedures/Functions/Methods

Performance Skill

Students will use algorithms and programming to solve problems and evaluate the results.

STRAND 4

The Internet
The Internet is built on systems that use protocols to transfer data.

Standard 1

The Internet

- Explain how computing devices work together in a network (Network, Path, Routing, Packets, Bandwidth).
- Explain how the Internet works (Fault Tolerance, Protocols, HTTP, HTTPS).
- Understand the difference between the Internet and the World Wide Web.

Standard 2

Web Development
Students will understand that the HTML programming language is used to create all websites on the internet and acts as the structure for a website.

- Students will code the foundation for a basic webpage including the element tags <!DOCTYPE html>, <html>, <head>, <title>, and <body>.
- Students will create pages with tags and attributes at the inline level. (<!DOCTYPE html>, <html>, <head>, <title>, <body>, <h1>, <h2>, <h6>, <p>, <br>, etc.)

Performance Skill

Students will explain how computer systems and networks, primarily the Internet, work. Students will design a simple webpage.

STRAND 5

Impact of Computing
The impact of computing extends to societal, economical, and cultural issues.

Standard 1

Beneficial and Harmful Effects

- Explore how an effect of a computing innovation can be both beneficial and harmful.
- Explore advances in computing that have generated and increased creativity in other fields, such as medicine, engineering, communications, and the arts.

Standard 2

Digital Divide and Computing Bias

- Explore issues that contribute to the digital divide (demographics, geographics, socioeconomic, equity, access, influence).
- Explore how bias exists in computing innovations.

Standard 3

Legal and Ethical Concerns

- Explain how the use of computing can raise legal and ethical concerns.
- Understand how ease of access and distribution of digitized information raises intellectual property concerns regarding ownership, value, and use.
- Understand the differences between Copyright, Creative Commons, Public Domain, & Trademark

Standard 4

Safe Computing

- Describe the risks to privacy from collecting and storing personal data on a computer system.
- Explain how computing resources can be protected (password strength) and can be misused.
- Explain how unauthorized access to computing resources is gained.
- Understand essential cybersecurity concepts.
  - Malware (adware, trojan horse, virus, ransomware, etc.)
  - Social Engineering (phishing, etc.)

Performance Skills

Students will discuss the benefits and issues of computing.
Students will develop and use safe internet practices.

Workplace Skills

Students will develop professional and interpersonal skills needed for success in the industry.

- Understand the difference between Hard Skills and Soft Skills.
- Identify Soft Skills needed in the workplace:
  - Good communication
  - Ability to problem solve
  - Teamwork
  - Critical Thinking
  - Dependability
  - Accountability
  - Respect legal requirements and expectations

Skill Certificate Test Points by Strand

Test Name	Test #	Number of Test Points by Strand										Total Points	Total Questions
		1	2	3	4	5	6	7	8	9	10		
Computer Science Principles	803	7	12	13	6	8						46	36

Computer Science Principles Vocabulary

Strand 1 - Creative Development	
Top-Down Design	A problem-solving approach in which you break a large problem into smaller pieces.
Debugging	Finding and fixing problems in an algorithm or program.
Logic Error	A logical mistake in the code that produces incorrect output.
Syntax Error	Errors which are detected and prevent the program from running.
Run Time Error	An error in the code that occurs while the program is running.
Strand 2 - Computing and Data	
Hardware	Physical components of a computer
Software	A program or a collection of programs
Input	The information computers get from users, devices, or other computers
Output	The information computers give to users, devices, or other computers
Storage	Saving information to use in the future
Bit	A single binary digit represented as a 0 or 1.
Byte	8 bits = 1 byte
Kilobyte	Approximately 1 thousand bytes
Megabyte	Approximately 1 million bytes
Gigabyte	Approximately 1 billion bytes
Terabyte	Approximately 1 trillion bytes
Petabyte	Approximately 1 quadrillion bytes
Processing	An action a computer performs when it receives data
Binary	A base-2 numbering system that computers use to process data
Decimal	A base-10 numbering system that humans typically use to process data
Lossless Compression	Compression algorithm in which file size is reduced without any quality loss.
Lossy Compression	Compression algorithm in which some of the data from the original file is lost.
Strand 3 - Algorithms and Programming	
Algorithm	A finite set of instructions that accomplish a task.
Assignment Operator	Used to change the value of a variable. Example: score = 10
Variable	A named value within a program.
String	An ordered sequence of characters.
Boolean Value	A data type that is either true or false.
Arithmetic operators	Includes addition, subtraction, multiplication, division, and modulus operators.
Modulus	An operator that returns the remainder of division
Comparison Operators	<, >, ≤, ≥, ==, ≠ indicate a Boolean expression
Logical operator	NOT, AND, and OR, which evaluate to a Boolean value.
Conditional Statement	Decision making based on a Boolean value (IF, ELSE IF, ELSE).
Concatenation	Joins together two or more strings to make a new string.
Iteration/loops	Part of a program that repeats a specified number of times (FOR loop) or until a given condition is met (WHILE loop).
Procedure/Functions/Methods	A named group of programming instructions.
Procedure/Function/Method Call	A command that executes a procedure, function, or method.

Strand 4 - The Internet	
Network	The hardware that connects computers together to transfer data.
Internet	A global network of networks.
World Wide Web	A collection of information that is accessed via the internet such as web pages or the cloud.
Path	The connections between devices on a network such as ethernet, fiber optics, WiFi, and cellular.
Packet	A chunk of data sent over a network.
Bandwidth	The maximum amount of data that can be sent in a fixed amount of time, usually measured in bits per second.
Router	A type of computer that forwards data across a network
Fault Tolerance	A network's ability to continue operating uninterrupted despite the failure of one or more of its components
Protocol	An agreed-upon set of rules for transmitting data.
HTTP	HyperText Transfer Protocol - the protocol used for transmitting web pages over the Internet
HTTPS	HyperText Transfer Protocol Secure - the protocol used for transmitting encrypted web pages securely over the Internet
HTML	HyperText Markup Language - the language used to create a webpage.
HTML Tags	Code that modifies a webpage, such as <!DOCTYPE html>, <html>, <head>, <title>, <body>, <h1 - h6>, <p>, , etc.
Strand 5 - Impact of Computing	
Digital Divide	Differing access to computing devices and the Internet, based on socioeconomic, geographic, or demographic characteristics.
Data bias	An error in which certain elements of a data set do not accurately reflect the full population or phenomenon.
Malware	Short for "malicious software." Malware is software designed to cause damage.
Adware	Malware that automatically displays or downloads advertisements.
Trojan Horse	Malware which appears harmless and pretends to do something useful.
Virus	Malware that can copy itself and gain access to a computer in an unauthorized way
Ransomware	Malware that blocks access to a computer system until a "ransom" is paid.
Social Engineering	Manipulating people into performing certain actions or giving up confidential information.
Phishing	A type of social engineering attack that attempts to trick a user into providing personal information.
Copyright	An exclusive right that is applied to tangible works
Creative Commons	Works which copyright is reserved but can be copied
Public Domain	Works that are not protected by copyright
Trademark	Symbols or words that a company has exclusive rights to use

Computer Science Principles Skills Reference Sheet

Assignment, Display, and Input	
`a = expression`	Evaluates `expression` and then assigns a copy of the result to the variable **a**.
`DISPLAY(expression)`	Displays the value of `(expression)` in the console window.
`INPUT( )`	Accepts a value from the user and returns the input value.
Arithmetic Operators and Numeric Procedures	

`a + b`
`a - b`
`a * b`
`a / b`
	The arithmetic operators `+`, `–`, `*`, and `/` are used to perform arithmetic on `a` and `b`.
	For example, `17 / 5` evaluates to `3.4`.
	The order of operations used in mathematics applies when evaluating expressions.

Relational and Boolean Operators	
`NOT condition`	Evaluates to `true` if `condition` is `false`; otherwise evaluates to `false`.
`condition1 AND condition2`	Evaluates to `true` if both `condition1` and `condition2` are `true`; otherwise evaluates to `false`.
`condition1 OR condition2`	Evaluates to `true` if `condition1` is `true` or if `condition2` is `true` or if both `condition1` and `condition2` are `true`; otherwise evaluates to `false`.

`a MODULUS b`
`-or-`
`a MOD b`
	Evaluates to the remainder when `a` is divided by `b`.
	For example, `17 MOD 5` evaluates to `2`.
	`MODULUS (MOD)` has the same precedence as the `*` and `/` operators.

`FOR(condition)`
`{`
`    <block of statements>`
`}`
	The code in `<block of statements>` is executed a certain number of times.

`WHILE(condition)`
`{`
`    <block of statements>`
`}`
	The code in `<block of statements>` is repeated until the `(condition)` evaluates to `false`.

`IF(condition1)`
`{`
`    <first block of`
`statements>`
`{`
`ELSE IF(condition2)`
`{`
`    <second block of`
`statements>`
`}`
`ELSE`
`{`
`    <third block of`
`statements>`
`}`
	If `(condition1)` evaluates to `true`, the code in `<first block of statements>` is executed; if `(condition1)` evaluates to `false`, then `(condition2)` is tested; if `(condition2)` evaluates to `true`, the code in `<second block of statements>` is executed; if both `(condition1)` and `(condition2)` evaluate to `false`, then the code in `<third block of statements>` is executed.

Procedures and Procedure Calls	

`PROCEDURE procName( )`
`{`
`    <block of`
`statements>`
`}`
	Defines `procName` as a procedure that takes no arguments. The procedure contains `<block of statements>`.
	The procedure `procName` can be called using the following notation:
	`procName( )`