Secondary Curriculum Maps

Cumberland Valley School District

Soaring to Greatness, Committed to Excellence

Introduction to Computer Science

Grade: 9-12			Intro to Computer Science (3091)
Unit	**Timeline**	**Topics**	**Priority Standards**
History of Computers and Programming Languages + Number Bases	15 Days	Computer History (5 days)	* 1B-IC-18 -- Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices.
		Language History (5 Days)	* 3B-AP-24 -- Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems.
		Number Bases (5 Days)	* 3A-DA-09 --Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
			(supporting standards)*
Program Development	Concurrent (~135 Days)	Basic syntax for Python	3B-AP-10 -- Use and adapt classic algorithms to solve computational problems.
		Basic syntax for Visual BASIC	3B-AP-23 -- Evaluate key qualities of a program through a process such as a code review.
Documenting code in Python			
Coding Fluency	Concurrent (~135 Days)	Docmenting Code and Stylistic Guideliens in Python/Visual Basic	3B-AP-18 -- Explain security issues that might lead to compromised computer programs.
		Debugging Code in Python and Visual Basic	
Programming Structures	135 Days	Input/Output in Python (15 days)	3B-AP-11 -- Evaluate algorithms in terms of their efficiency, correctness, and clarity.
		Conditionals (10 days)	3B-AP-12 -- Compare and contrast fundamental data structures and their uses.
		Loops in Python (10 days)	3B-AP-14 -- Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
		Arrays, List, Dictionaries in Python (10 days)	
		Strings (10 days)	
		Functions in Python (10 days)	
		Input/Output in Visual BASIC (10 days)	
		Loops in Visual BASIC (10 days)	
		Arrays in Visual BASIC (10 days)	
		Subroutines in Visual BASIC (10 days)	
		Objects in Visual BASIC (30 days)	

Computer Science Curriculum Map

<table>
<tr><th colspan="2">CSTA K-12 Standards 2017 Revision</th></tr>
<tr><td colspan="2"><b>3B-AP-23 -- Evaluate key qualities of a program through a process such as a code review</b></td></tr>
<tr><th colspan="2">Taught in Unit(s)</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th colspan="2">Explanation/Example of Standard</th></tr>
<tr><td colspan="2">Examples of qualities could include correctness, usability, readability, efficiency, portability, and scalability.</td></tr>
<tr><th colspan="2">Common Misconceptions</th></tr>
<tr><td colspan="2"><b>Lack of commenting or no commenting at all</b>
<b>Poor spacing</b>
<b>No header</b></td></tr>
<tr><th>Big Idea(s)</th><th>Essential Question(s)</th></tr>
<tr><td><b>When a program doesn't work, there are ways to fix it.</b>
<b>Your programming peers need to understand your code!</b></td><td><b>How do I identify and debug any errors in my program?</b>
<b>How do I enter, document, and execute a simple program?</b></td></tr>
<tr><th colspan="2">Assessments</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>Concepts
(what students need to know)</th><th>Skills
(what students must be able to do)</th></tr>
<tr><td>What are the qualities of a well-documented, efficient program which follow a set of stylistic guidelines.</td><td>Students will be able to evaluate a program on its efficiency, correctness, and readability.</td></tr>
</table>

Computer Science Curriculum Map

CSTA K-12 Standards 2017 Revision	
3B-AP-18 -- Explain security issues that might lead to compromised computer programs	
Taught in Unit(s)	
Explanation/Example of Standard	
For example, common issues include lack of bounds checking, poor input validation, and circular references.	
Common Misconceptions	
Assuming a program works simply because they tried one correct test case **Traversing too far through an array** **Using the incorrect variable type - integers versus decimals**	
Big Idea(s)	**Essential Question(s)**
When a program doesn't work, there are ways to fix it.	**How do I identify and debug any errors in my program?**
Assessments	
Concepts (what students need to know)	**Skills** (what students must be able to do)
Understand and identify errors in programming syntax to explain common issue(s) with the code.	Students will be able to identify explain common syntax and logic errors in code.

Computer Science Curriculum Map

<table>
<tr><th colspan="2">CSTA K-12 Standards 2017 Revision</th></tr>
<tr><td colspan="2"><b>3B-AP-14 -- Construct solutions to problems using student-created components, such as procedures, modules, and/or objects.</b></td></tr>
<tr><th colspan="2">Taught in Unit(s)</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th colspan="2">Explanation/Example of Standard</th></tr>
<tr><td colspan="2">Object-oriented programming and other problems which can be assigned or student-selected.</td></tr>
<tr><th colspan="2">Common Misconceptions</th></tr>
<tr><td colspan="2"><b>Putting too much into a single procedure</b>
<b>Overusing global variable rather than passing variables as parameters</b>
<b>Misunderstanding the nature of timers in Visual BASIC</b></td></tr>
<tr><th>Big Idea(s)</th><th>Essential Question(s)</th></tr>
<tr><td><b>There is an optimal approach and an efficient method to unpack assigned tasks.</b>
<b>Coding applies beyond the classroom!</b>
<b>As coding languages are robust, programmers should have the ability to research/explore topics which are new or unknown.</b></td><td><b>How do I write a series of programming instructions in a logical sequence to solve a problem?</b>
<b>What are some resources I can use to enhance my knowledge of coding beyond the scope of this class?</b></td></tr>
<tr><th colspan="2">Assessments</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>Concepts
(what students need to know)</th><th>Skills
(what students must be able to do)</th></tr>
<tr><td>How are subroutines, functions, and procedures constructed and added into program.</td><td>Students will be able to construct subroutines, functions, and procedures using prior knowledge as well as be able to research keywords and concepts beyond the scope of the class.</td></tr>
</table>

Computer Science Curriculum Map

<table>
<tr><th colspan="2">CSTA K-12 Standards 2017 Revision</th></tr>
<tr><td colspan="2"><b>3B-AP-12 -- Compare and contrast fundamental data structures and their uses.</b></td></tr>
<tr><th colspan="2">Taught in Unit(s)</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th colspan="2">Explanation/Example of Standard</th></tr>
<tr><td colspan="2">Examples could include strings, lists, arrays, stacks, and queues.</td></tr>
<tr><th colspan="2">Common Misconceptions</th></tr>
<tr><td colspan="2"><b>Using the wrong data structure
Type mismatch between strings and numbers
Using the wrong index on a structure (for example, not starting at 0)</b></td></tr>
<tr><th>Big Idea(s)</th><th>Essential Question(s)</th></tr>
<tr><td><b>There is an optimal approach and an efficient method to unpack assigned tasks.
Coding applies beyond the classroom!</b></td><td><b>How do I write a series of programming instructions in a logical sequence to solve a problem?
What are looping structures and how do they improve our programs?
What are arrays/lists and how do they improve our programs?
What are subroutines/functions and how do they improve our programs?
What are strings and what are some functions which we can use in our programs to manipulate them?</b></td></tr>
<tr><th colspan="2">Assessments</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>Concepts
(what students need to know)</th><th>Skills
(what students must be able to do)</th></tr>
<tr><td>What are the various coding data structures, and what are the similarities and differences between them?</td><td>Students will be able to differentiate between the various data structures such as loops, arrays, strings, functions, etc...</td></tr>
</table>

Computer Science Curriculum Map

CSTA K-12 Standards 2017 Revision	
3B-AP-11 -- Evaluate algorithms in terms of their efficiency, correctness, and clarity.	
Taught in Unit(s)	
Explanation/Example of Standard	
Examples could include sorting and searching.	
Common Misconceptions	
Just because the program works doesn’t mean it’s the most efficient way to solve the task. **Repeatedly coding something rather than using a single subroutine** **Miscounting the number of steps an algorithm takes to execute**	
Big Idea(s)	**Essential Question(s)**
There is an optimal approach and an efficient method to unpack assigned tasks. **When a program doesn’t work, there are ways to fix it.**	**How do I write a series of programming instructions in a logical sequence to solve a problem?** **How do I identify and debug any errors in my program?**
Assessments	
Concepts (what students need to know)	**Skills** (what students must be able to do)
How are algorithms evaluated for their efficiency, correctness, and clarity	Students will be able to determine the efficiency, correctness, and clarity of algorithms by testing and documenting their code..

Computer Science Curriculum Map

<table>
<tr><th colspan="2">CSTA K-12 Standards 2017 Revision</th></tr>
<tr><td colspan="2"><b>3B-AP-10 -- Use and adapt classic algorithms to solve computational problems.</b></td></tr>
<tr><th colspan="2">Taught in Unit(s)</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th colspan="2">Explanation/Example of Standard</th></tr>
<tr><td colspan="2">Examples could include sorting and searching.</td></tr>
<tr><th colspan="2">Common Misconceptions</th></tr>
<tr><td colspan="2">The order of lines of code (For example, calculating a formula before the user enters input.)
Assignment dyslexia (x + 6 = x rather than x = x + 6)
Improper logic checking (For example, multiple if rather than if/elseif.)</td></tr>
<tr><th>Big Idea(s)</th><th>Essential Question(s)</th></tr>
<tr><td><b>There is an optimal approach and an efficient method to unpack assigned tasks.</b>
<b>Coding applies beyond the classroom!</b></td><td><b>How do I write a series of programming instructions in a logical sequence to solve a problem?</b></td></tr>
<tr><th colspan="2">Assessments</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>Concepts
(what students need to know)</th><th>Skills
(what students must be able to do)</th></tr>
<tr><td>How are programming keywords and syntax used to solve computational problems.</td><td>Students will be able to use the proper programming keywords and syntax to solve computational problems.</td></tr>
</table>