SmashCode #1

Course Description: SmashCode #1 introduces students to the exciting world of coding via the Python Turtle Graphics programming library. This computer science course will introduce students to the Python Turtle Module, emphasizing the movement of objects around an environment and the creation and manipulation of shapes. Students will learn the foundations of computer science and the basics of programming, with an emphasis on helping students develop logical thinking and problem-solving skills. SmashCode #1 will also provide a smooth transition into SmashCode #2 and is highly recommended for students considering a computer science-related field.

Grades: 9-12

Course Resources & Materials: CodeHS Intro to Computer Science in Python 3

Course Essential Questions:

- How can we use computer programming to model and solve real world situations and problems?
- What makes a good program?
- How can computing and the use of computational tools foster creative expression?
- What does it mean to be literate in the 21st century?

Course Priority Standards:
DESE
MO.9-10.AP.V.01 Create problem solutions that utilize primitive variables (e.g., strings, ints, booleans, doubles).
MO.9-10.AP.C.01 Apply the concepts of specific control structures (e.g., sequence, conditionals, repetition, procedures).considering program efficiencies such as readability, performance and memory usage.
MO.11-12.AP.M.01 Construct solutions to problems using student-created components (e.g., procedures, modules, objects).
MO.11-12AP.PD.05 Develop and use a series of test cases to verify that a program performs according to its design specifications.

CSTA
2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.
3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
Course Learning Goals and Objectives:

- Students will move an object across a coordinate plane along the X and Y axis.
- Students will use a For Loop to repeat a pattern a specific number of times.
- Students will build functions to simplify code.
- Students will use Top Down Design to break large problems into smaller solvable problems.
- Students will use variables as units of storage.
- Students will take User Input to allow interaction with a program.
- Students will use If/Else statements to branch to different chunks of code.
- Students will use all concepts learned in this course to create a Choose-Your-Own-Adventure Project using the Python Turtle Module.

Unit 1	**Intro to Programming with Turtle Graphics (10 Weeks)** In this unit, students learn about the variety of code available via the Python Turtle Module and maneuver an object around a fixed environment.
Unit 1 Enduring Understandings	• Algorithms are written in order to provide instruction to a program in the form of commands. In turn, commands are used to move an object throughout an environment based on a coordinate system. • Programmers write commands to move, draw and interact with an environment. • For Loops are used to repeat a command a specific number of times. • Functions make programs more efficient by allowing for reusable chunks of code to be recalled. • Top Down Design allows large problems to be broken into smaller more manageable pieces. • User Input can be stored in variables and in If/Else Statements to branch to different code segments.
Unit 1 Essential Questions	• How can we use computer programming to model and solve real-world situations and problems? • How do computers help people to solve problems? • How do people and computers approach problems differently? • What does a computer need from people in order to solve problems effectively? • What is a Command? • How can we move an object around a fixed coordinate plane? • What is a For Loop? • What are Functions and Parameters? • What is Top Down Design? • How do Variables work? • How is User Input created and stored? • How do If/Else Statements branch? • What is a While Loop and how does it differ from a For Loop?
Unit 1 Student Learning Goals	• Students will write a program that will have an object draw a stretched-out slinky. • Students will use the backward, penup pendown commands to draw a caterpillar on their canvas. • Students will use for loops to enhance to draw and enhance a 4 Columns' program.

	● Students will use the object to create a pyramid of circles with 3 circles on the bottom row, 2 on the middle row, and 1 on the top. ● Students will add comments to a previously built program. ● Students will use the object to draw a stack of squares and circles from the bottom of the canvas to the top using iteration. ● Students will write a program that will draw a line of 5 blocks that increase in size using Top Down Design. ● Students will write a program that has the object draw a square in each corner of the canvas. ● Students will write a program that draws a happy or sad face based on the user's mood using If/Else statements. ● Students will write a program that will draw a red and black checkerboard on the canvas iteration and modulus operator.
Unit 1 Vocabulary	Turtle Module, grid, setposition, for loop, comment, functions, Top Down Design, input, iteration, if/else, while loop
Unit 1 DESE Missouri Learning Standards Computer Science Standards K-12	**MO.9-10.AP.V.01** Create problem solutions that utilize primitive variables (e.g., strings, ints, booleans, doubles)
	MO.9-10.AP.C.01 Apply the concepts of specific control structures (e.g., sequence, conditionals, repetition, procedures) considering program efficiencies such as readability, performance and memory usage.
	MO.9-10.AP.M.01 Break down a solution into procedures using systematic analysis and design utilizing functional abstraction.
	MO.9-10.AP.PD.04 While collaborating in a team, develop, test and refine programs that solve practical problems or allow self-expression.
	MO.11-12.AP.C.01 Trace the execution of iteration (e.g., loops, recursion), illustrating output and changes in values of named variables.
	MO.11-12.AP.M.01 Construct solutions to problems using student-created components (e.g., procedures, modules, objects).
	MO.11-12.AP.PD.05 Develop and use a series of test cases to verify that a program performs according to

	its design specifications.
Unit 1 Computer Science Teacher Association K-12 Standards	**2-AP-11** Create clearly named variables that represent different data types and perform operations on their values.
	2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
	2-AP-14 Create procedures with parameters to organize code and make it easier to reuse.
	2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.
	3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
	3B-AP-12 Compare and contrast fundamental data structures and their uses.
	2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.
	2-AP-19 Document programs in order to make them easier to follow, test, and debug.
	3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
	3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
	3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
	3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
	3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

	3B-AP-12 Compare and contrast fundamental data structures and their uses.
	3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
	3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
Unit 1 Assessments	Each individual unit subcategory has one summative Check for Understanding as well as multiple formative options to demonstrate skills.
Curricular Resources Utilized in Unit 1	CodeHS Intro to Computer Science in Python 3

Unit 2	**Basic Python and Console Interaction (8 Weeks)** In this unit, students will write programs that focus on console interaction.
Unit 2 Enduring Understandings	• Algorithms are written in order to provide instruction to a program in the form of commands. In turn, commands are used to allow console interaction. • Programmers write commands that allow a user to interact with a program via the console. • The print command allows text to be displayed via the console. • Variables allow for information to be stored. • Variable types determine the type of element. • The input command allows a programmer to get input from the user. • Mathematical Operators allow programs to perform.
Unit 2 Guiding & Essential Questions	• How can we use computer programming to model and solve real world situations and problems? • How do computers help people to solve problems? • How do people and computers approach problems differently? • What does a computer need from people in order to solve problems effectively? • What is a Command?

	● How can we move an object around a fixed coordinate plane? ● What is a For Loop? ● What are Functions and Parameters? ● What is Top Down Design? ● How do Variables work? ● How is User Input created and stored? ● How do If/Else Statements branch? ● What is a While Loop and how does it differ from a For Loop?
Unit 2 Student Learning Goals	● Students will write a program that prints their name and something about themselves. ● Students will write a program that displays their name written vertically. ● Students will write a program that does the following: creates a string variable, creates an integer variable, and prints both variables, each on its own line. ● Students will write a program that asks the user how old they are, and their age, and prints the precise number of candles needed for a birthday cake. ● Students will write some mathematical expressions to compute the area and perimeter of the rectangle and save these values inside variables named area and perimeter. ● Students will write a program that asks the user for the following things: names of three ingredients for a recipe, the number of ounces of each ingredient required for one serving, and the number of servings desired. ● Students will add a multi-line comment at the top of a program, describing the program's function. ● Students will add a small single-line comment above each contiguous block of code, describing what that block does.
Unit 2 Vocabulary	print, variable, variable type, string, integer, float, input, comments
Unit 2 DESE Missouri Learning Standards Computer Science Standards K-12	**MO9-10.AP.V.01** Create problem solutions that utilize primitive variables (e.g., strings, ints, booleans, doubles).
	MO9-10AP.C.01 Trace the execution of iteration (e.g., loops, recursion), illustrating output and changes in values of named variables.
	MO11-12AP.C.01 Apply the concepts of specific control structures (e.g., sequence, conditionals, repetition, procedures) considering program efficiencies such as readability, performance and memory usage.

Unit 2 Computer Science Teacher Association K-12 Standards	**2-AP-11** Create clearly named variables that represent different data types and perform operations on their values.
	2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
	2-AP-19 Document programs in order to make them easier to follow, test, and debug.
	2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.
	3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
	3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
	1B-AP-15 Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.
	1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.
	1A-AP-14 Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
	1B-AP-08 Compare and refine multiple algorithms for the same task and determine which is the most appropriate.
	1B-AP-10 Create programs that include sequences, events, loops, and conditionals.
Unit 2 Assessments	Each individual unit subcategory has one summative Check for Understanding as well as multiple formative options to demonstrate skills.
Curricular Resources Utilized in Unit 2	CodeHS Intro to Computer Science in Python 3