Intro to Computer Science with Python Scope and Sequence

The CodeHS introduction to Python course teaches the fundamentals of computer programming as well as some advanced features of the Python language. Students use what they learn in this course to build simple console-based games. This course is equivalent to a semester-long introductory Python course at the college level.

Module 1: Introduction to Programming with Tracy the Turtle	
15 hours (3 weeks)	Students are introduced to the Python programming language by learning to use various commands and control structures to write programs that will create graphical designs.

CSTA Standards Addressed

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.

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-18 Create artifacts by using procedures within a program, combinations or data and procedures, or independent but interrelated programs.

Module 2: Basic Python and Console Interaction	
15 hours (3 weeks)	Students learn the basics of programming by writing programs that incorporate interaction using a keyboard.

CSTA Standards Addressed

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 3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.

Module 3: Control Flow	
50 hours (10 weeks)	Students learn how to teach their programs to make decisions based on the information it receives. They will learn how to decompose programs into smaller pieces that work together to solve a problem.
CSTA Standards Addressed 2-AP-11 Create clearly names variables that represent different data types and perform operations on their values. 2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 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, presentation, and/or demonstrations in the development of complex programs. 3B-AP-12 Compare and contrast fundamental data structures and their uses.	

Module 4: Strings	
15 hours (3 weeks)	Students learn more sophisticated strategies for manipulating text in programs - slicing, concatenating, and formatting.
CSTA Standards Addressed 2-AP-11 Create clearly names 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-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. 3A-AP-23 Document design decisions using text, graphics, presentation, and/or demonstrations in the development of complex programs. 3B-AP-12 Compare and contrast fundamental data structures and their uses.

Module 5: Data Structures	
30 hours (6 weeks)	Students will build more complex programs that make use of lists, grids, and dictionaries.
CSTA Standards Addressed 2-AP-11 Create clearly names variables that represent different data types and perform operations on their values. 2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. 3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables. 3A-AP-23 Document design decisions using text, graphics, presentation, and/or demonstrations in the development of complex programs. 3B-AP-12 Compare and contrast fundamental data structures and their uses.	

Module 6: Project- Hangman	
15 hours (3 weeks)	Students use the skills they've learned in the first four modules to build the game Hangman.
CSTA Standards Addressed	

2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

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-18 Create artifacts by using procedures within a program, combinations or data and procedures, or independent but interrelated programs.

3A-AP-23 Document design decisions using text, graphics, presentation, and/or demonstrations in the development of complex programs.

3B-AP-12 Compare and contrast fundamental data structures and their uses.

Supplemental: Module 7: Classes and Objects	
30 hours (6 weeks)	Students learn the principles of object-oriented design.

CSTA Standards Addressed

2-AP-11 Create clearly names variables that represent different data types and perform operations on their values.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing

computational artifacts.

3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-23 Document design decisions using text, graphics, presentation, 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-centered component, such as procedures, module, and/or objects.

3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.

Supplemental: Module 8: Project- Who Said it?	
15 hours (3 weeks)	This final project allows students to combine a variety of topics in a single program, utilizing file reading, top down design, and debugging skills.

CSTA Standards Addressed

2-AP-11 Create clearly names variables that represent different data types and perform operations on their values.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

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 or data and procedures, or independent but interrelated programs.

3A-AP-23 Document design decisions using text, graphics, presentation, 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-centered component, such as procedures, module, and/or objects.

3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.