TechSmart

CS201: Coding in Python 1

Course Syllabus

Course Description

This course provides an in-depth introduction to coding in Python. Upon completion, students will master fundamental coding concepts such as statements, variables, expressions, conditionals, and loops. Students will also gain proficiency with advanced topics including software libraries, automation, and sprite-based graphics.

In addition, students will learn how to plan and track the progress of large coding projects, debug errors, and improve the readability of their code. Throughout the course, students will continuously demonstrate their knowledge through both traditional assessments and coding projects such as games, animations, and other interactive programs.

Course Outline

Unit 1: Linear Programs	Unit 2: Decisions	Unit 3: Loops
Lesson 1.1 Statements and Variables	Lesson 2.1 Conditionals (if)	Lesson 3.1 While Loops
Lesson 1.2 Libraries	Lesson 2.2 Conditionals (elif and else)	Lesson 3.2 Controlling Loops
Lesson 1.3 Values	Industry Practice Code Style	Lesson 3.3 Classes
Research Question Tech Impact	Unit 2 Quiz	Lesson 3.4 Graphics
Lesson 1.4 Expressions	Research Question Automation	Unit 3 Quiz
Unit 1 Test	Industry Practice Debugging	Research Question Intellectual Property
Industry Practice Planning a Program	Lesson 2.3 Built-in Libraries	Lesson 3.5 Animation
	Lesson 2.4 Booleans	Lesson 3.6 Interaction
	Unit 2 Test	Lesson 3.7 For-Range Loops
	Industry Practice Scoping and Presenting Work	Unit 3 Test
	Unit 2 Project	Industry Practice Kanban
		Unit 3 Project

TechSmart

Learning Objectives

Unit 1: Linear Programs	1.1: Statements and Variables	
	Comprehension Objectives	• Define the lesson terms • Describe how a computer executes code • Identify input and output in a program • Identify variables and their values • Identify, describe, and differentiate between camelCase notation and underscore_notation for variable names
	Application Objectives	• Use basic console text input and output commands • Store values in variables • Combine strings (both variables and literals) using the + operator • Debug common problems related to lesson topics
	1.2: Libraries	
	Comprehension Objectives	• Define the lesson terms • Describe what happens in the flow of code when a function is called • Identify function calls in code • Identify the arguments sent to a function • Explain why arguments may be necessary for functions • Use documentation to identify what arguments are necessary for a given function
	Application Objectives	• Import a library • Call a function from a library using the correct arguments • Debug common problems related to lesson topics
	1.3: Values	
	Comprehension Objectives	• Define the lesson terms • Identify which data type is most appropriate for a given situation • Distinguish between literals and variables

TechSmart

	• Identify the data type of a given value • Identify when a basic mathematical operator will produce an integer and when it will produce a float • Give the order of operations for basic mathematical operators
Application Objectives	• Combine strings (both variables and literals) using the + operator • Use typecasting to temporarily alter the type of a value • Use basic mathematical operators on integers and floats: +, -, /, * • Create a printable string value by combining strings and numbers • Debug common problems related to lesson topics
Research Question: Tech Impact	
Comprehension Objectives	• Define the lesson terms • Give examples of search strategies that could be used to research the lesson topic • Explain why citing sources is important • Learn about careers that use computer science • Describe how technology has changed culture over time
Application Objectives	• Gather information from a variety of sources • Evaluate the accuracy and bias of sources • Provide citations for sources used • Evaluate how technology has impacted various career fields • Consider how technology might impact a career field in the future
1.4: Expressions	
Comprehension Objectives	• Define the lesson terms • Given a statement that uses a compound assignment operator, give the full version of the statement with separate assignment and math operators and vice-versa • List the benefits of using compound assignment

		operators ● Identify an expression within a line of code ● Distinguish between a statement (performs a complete action) and an expression (produces a value, but does nothing with it) ● Identify when a command (such as input) is both an expression and a statement ● Identify when a function may be treated as an expression (e.g. when it returns a value) ● Describe what happens in the flow of code when a function with a return value is called ● Use documentation to identify whether a function returns a value that may be stored ● Identify 'None' as the value returned by any function that does not have an explicit return value
	Application Objectives	● Typecast an input expression to produce a number result ● Predict the result of an expression ● Use compound assignment operators (+=. -=. *=, \=) ● Describe where these operators fall in the order of operations for Python ● Store the result of a function ● Use a function as part of an expression ● Use the following techniques: Typecasting Input, Incrementing a String ● Debug common problems related to lesson topics
	Industry Practice: Planning a Program	
	Comprehension Objectives	● Define the lesson terms ● Describe the product life cycle as a formal process for creating software ● Identify and describe "Envision" and "Design" as the first two steps in the product life cycle ● Explain why a planning phase is necessary and useful before beginning a larger project ● Differentiate between pseudocode and true syntax

	Application Objectives	● Outline a given program as pseudocode ● Break a given program / problem down into smaller features / sub-problems ● Brainstorm ideas and present the result as a list of detailed features ● Translate a feature list into a pseudocode outline ● Implement a program from a pseudocode outline

Unit 2: Decisions	2.1: Conditionals (if)	
	Comprehension Objectives	● Define the lesson terms ● Identify if statements in code ● Identify the condition within an if statement ● Describe how an if statement makes a decision ● Identify which comparison operator is most appropriate in a given context ● Describe where comparison operators fall in the order of operations for Python ● Differentiate between the "=" and "==" operators and describe the function of each ● Explain how whitespace is used to delineate the beginning and end of conditional sections
	Application Objectives	● Write an if statement to make a decision ● Predict which code within a conditional will execute from looking at a program ● Use the following techniques: User Choice, Running Total, Limit Number ● Debug common problems related to lesson topics
	2.2: Conditionals (elif and else)	
	Comprehension Objectives	● Define the lesson terms ● Describe the flow of a conditional with elif- and else-clauses ● Describe the general format of a clause (i.e. begins with a keyword and ends with a ':') ● Identify when a conditional structure is nested ● Identify the range described by the 'min < num <

	max' chained comparison format
Application Objectives	● Predict which code within a conditional will execute from looking at a program ● Predict which conditionals within a nested structure will execute from looking at a program ● Write a conditional to make a decision between multiple cases ● Determine whether a number falls into a range defined by the min < num < max format ● Debug common problems related to lesson topics
Industry Practice: Code Style	
Comprehension Objectives	● Define the lesson terms ● Describe the benefits of good code style and commenting ● Identify "PEP-8" as the commonly accepted Python code style guidelines ● Explain that code style does not affect program output or functionality ● Differentiate between standard and header comment syntax
Application Objectives	● Improve the readability of programs using an good code style ● Improve the readability of programs using an appropriate level of comment density
Research Question: Automation	
Comprehension Objectives	● Describe how technology has changed culture over time
Application Objectives	● Make predictions about future technology based on existing technology ● Evaluate how technology has impacted various career fields
Industry Practice: Debugging	
Comprehension Objectives	● Define the lesson terms ● Identify the line number within an error message

TechSmart

		• Give examples of scenarios where line numbers may not be accurate (e.g. missing parentheses) • Give an approximate plain English translation of an error message • Identify and describe two different code-debugging strategies: ○ Using print statements ○ Reading the error message
	Application Objectives	• Choose which debugging strategy is most effective for a given situation • Use both strategies to find and fix errors
	2.3: Built-In Libraries	
	Comprehension Objectives	• Define the lesson terms • Identify and describe the random and math libraries • Give examples of commands found in the random and math libraries • Explain how to use documentation to find a full list of available commands in a library • Give examples of how randomness may be used in a program • Explain the relationship between randomness and Artificial Intelligence • List common programming uses for the modulus operator
	Application Objectives	• Use the random.randint() function • Use the modulus operator (%) • Use the exponentiation operator (**) • Use the floor division operator (//) • Use advanced math operations from the math library (such as sqrt) • Find and use other operations from math and random without explicit introduction to them • Use the following techniques: Random Choice, Weighted Choice • Debug common errors related to the lesson topics

	2.4: Booleans	
	Comprehension Objectives	● Define the lesson terms ● Contrast between the logical operators ● Identify boolean expressions in code ● Recognize comparison operators as operators that produce booleans ● Explain that booleans can be stored in variables like other data types ● Differentiate between well-formatted boolean variable conditions and redundant (bad boolean zen) versions
	Application Objectives	● Choose which logical operator is appropriate to combine values in a given situation ● Predict the values that will result from given boolean expressions ● Given a set of constraints or conditions under which something will happen, translate this into a compound boolean expression ● Simply a complex compound boolean expression by replacing various expressions with variables ● Reduce a bad boolean zen condition to a simpler form ● Use a single boolean variable as a condition ● Use the following techniques: Flexible Input, Reduce Compound Expressions ● Debug common problems related to lesson topics
	Industry Practice: Scoping and Presenting Work	
	Comprehension Objectives	● Define the lesson terms ● Explain why proper scoping for a project is important ● Differentiate between a modular approach and other, more monolithic approaches ● Explain how a modular approach allows for scoping up or down
	Application Objectives	● Choose a project of reasonable scope for a given time-frame ● Create a meaningful presentation of a program,

		explaining points of interest ● Present failures as well as successes as a normal part of a retrospective ● Divide a project into multiple releases or versions

Unit 3: Loops	3.1: While Loops	
	Comprehension Objectives	● Define the lesson terms ● Describe the logical flow of a loop ● Explain the importance of changing the loop condition inside the loop (e.g. avoiding infinite loops) ● List benefits of using loops (simplify code, run until signalled to stop, etc.)
	Application Objectives	● Looking at a loop, determine how much it will repeat / when it will stop ● Use while loops to repeat code until the user chooses to stop ● Create loops that are governed by a single boolean control variable ● Use the following techniques: Force Correct Input, Nested Loops, Player Turns, True Until False ● Debug common problems related to lesson topics
	3.2: Controlling Loops	
	Comprehension Objectives	● Define the lesson terms ● List and describe different variations on while loops (while, loop else clause) ● Differentiate between loops that end normally and loops that end with break ● Differentiate between the effect of 'break' and 'continue' within a loop block ● Describe alternatives to using a 'continue' statement (e.g. using conditionals to decide whether to do part of the loop block) ● Identify situations where it would be reasonable to use break

	Application Objectives	• Use break to exit a loop early • Use continue to skip skip the remainder of a loop iteration • Debug common problems related to lesson topics
	3.3: Classes	
	Comprehension Objectives	• Define the lesson terms • Describe how an instance is related to a class • Describe how methods and attributes are related to a class • Give examples of classes
	Application Objectives	• Set and get fields on an instance • Call methods of an instance • Use documentation to get information about the attributes and methods of a class without prior instruction on them • Use the technique: Change an Instance With a Function • Debug common problems related to lesson topics
	3.4: Graphics	
	Comprehension Objectives	• Define the lesson terms • Identify the arguments required to create various visual objects (window, sprite, etc.) • Describe how the main loop is used to keep a program open • Describe the conditions necessary to open, update, and close a window (e.g. the main loop using the is_running field, and the window.finish_frame command) • Identify what happens when you forget the window.finish_frame command (e.g. infinite loop) • Describe how coordinates are used to represent a position on-screen • List and describe different text alignments (left, center, right) • Contrast the programming coordinate space with the math coordinate space • Explain the value of labeling constants

TechSmart

		• Contrast variables and constants • Identify a constant based on the style conventions of its name (e.g. in ALL_CAPS)
	Application Objectives	• Open a window using tsapp • Draw static sprites with tsapp • Draw text to screen with tsapp • Given a set of coordinates and a window size, roughly estimate the coordinate position • Use the following techniques: Precise Positioning, Draw Order • Debug common problems related to lesson topics
	Research Question: Intellectual Property	
	Comprehension Objectives	• Explain, compare, and debate the effects of intellectual property laws • Explain the necessity of providing attribution, and the effects of failing to do so
	Application Objectives	• Debate laws and regulations that impact the development and use of software • Compare and evaluate licenses for different types of usage, including code licenses
	3.5: Animation	
	Comprehension Objectives	• Define the lesson terms • Describe how animation occurs because of rapid change in each iteration of a loop • Describe a sprite's speed as the number of pixels it moves in one second • Compare and contrast x vs y speeds • Compare and contrast positive vs negative speeds • Describe how a spritesheet is transformed into an animated image • Contrast between movement-based animation and image-based animation
	Application Objectives	• Perform simple animations by moving objects in a loop • Use an animated sprite based on a spritesheet

		• Animate a change in a sprite by manually changing the image • Change the speed of a sprite's visual or movement-based animation • Use the following techniques: Change Animation Rate, Change Direction • Debug common problems related to lesson topics
	3.6: Interaction	
	Comprehension Objectives	• Define the lesson terms • Explain the difference between states (e.g. whether a button is down) and events (e.g whether a button was pressed on this frame)
	Application Objectives	• Call methods that check for the current state of keys and mouse (e.g. position, is_down) • Call methods that check for events related to keys and mouse (e.g. was_pressed) • Call methods that check for collision between mouse and sprite, or two sprites • Use the following techniques: Follow Mouse, Move with Arrow Keys • Calculate duration by subtracting two points in time • Assign multiple variables at once using comma syntax • Debug common problems related to lesson topics
	3.7: For-Range Loops	
	Comprehension Objectives	• Define the lesson terms • Differentiate between while and for loops • Describe how the value of the loop variable changes as the loop continues • Describe how any for-range loop could be written as a while loop • Give the default 'range' values when not overridden (e.g. '0' for start and '1' for step)
	Application Objectives	• Choose whether a for-loop or a while-loop is more appropriate for a given situation

		• Use a for-range loop to loop a specific number of times • Use for-range variations to count by amounts other than 1 • Use for-range variations to count backwards • Use continue, break and else with a for loop • Use the following techniques: Counting Down, Row of Sprites • Debug common problems related to lesson topics
	Industry Practice: Kanban	
	Comprehension Objectives	• Define the lesson terms • Explain the importance of tracking work for large projects • Describe how kanban is used to track work • Compare kanban to other simple forms of work-tracking, such as checklists
	Application Objectives	• Create a kanban board that tracks tasks in a large project • Choose appropriate lanes for the type of project being undertaken • Debug common problems related to lesson topics

Standards and Certifications

Upon completion of CS201: Coding with Python 1 and CS202: Coding with Python 2, students will be prepared to take the Certified Entry-Level Python Programmer (PCEP) certification exam.

Additionally, students who complete CS201, CS202, and CS203: Coding with Python 3 will be prepared to take the Microsoft Technology Associate (MTA): Introduction to Programming Using Python certification exam.

	All Units	
	CSTA Standards	• **3A-AP-13:** Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. • **3A-AP-15:** Justify the selection of specific control

		structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made. • 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-IC-26: Demonstrate ways a given algorithm applies to problems across disciplines. • 3B-AP-10: Use and adapt classic algorithms to solve computational problems. • 3B-AP-15: Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. • 3B-DA-07: Evaluate the ability of models and simulations to test and support the refinement of hypotheses.

	Unit 1: Linear Programs	
	CSTA Standards	• 3A-CS-01: Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. • 3A-DA-09: Translate between different bit representations of real-world phenomena, such as characters, numbers, and images. • 3B-AP-17: Plan and develop programs for broad audiences using a software life cycle process. • 3A-IC-24: Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
	PCEP Certification	• Fundamental concepts: syntax and semantics, Python keywords, instructions • Literals: integer, floating-point numbers, strings • Comments • The print() function

		• The input() function • Numeric operators: * / + - • String operators: * + • Assignments and shortcut operators • Basic input and output operations using the input(), print(), int(), float(), str() functions • Type casting • Basic calculations • Simple strings: constructing, assigning • The None keyword
	MTA Certification	• Evaluate an expression to identify the data type Python will assign to each variable • Identify str, int, float, and bool data types • Convert from one data type to another type • Select the appropriate operator to achieve the intended result • Assignment and arithmetic operators • Read input from console; print formatted text • Determine the sequence of execution based on operator precedence: assignment; arithmetic

	Unit 2: Decisions	
	CSTA Standards	• 3A-AP-23: Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. • 3B-AP-09: Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem. • 3B-IC-27: Predict how computational innovations that have revolutionized aspects of our culture might evolve.
	PCEP Certification	• Fundamental concepts: indenting • Literals: boolean • Numeric operators: ** % // • Boolean operators: not and or • Relational operators: == != > >= < <=, building

		complex boolean expressions ● Conditional statements: if, if-else, if-elif, if-elif-else ● Multiple conditional statements ● Nesting conditional statements
	MTA Certification	● Comparison and logical operators ● if; elif; else; nested and compound conditional expressions ● Document code segments using comments ● Use indentation, white space, comments ● Syntax errors; logic errors; runtime errors ● Math; random ● Determine the sequence of execution based on operator precedence: comparison; logical

	Unit 3: Loops	
	CSTA Standards	● 3A-AP-20: Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. ● 3B-IC-28: Debate laws and regulations that impact the development and use of software.
	PCEP Certification	● Building loops: while, for, range() ● Iterating through sequences ● Expanding loops: while-else ● Nesting loops
	MTA Certification	● while; for; break; continue; pass; nested loops and loops that include compound conditional expressions