Coding in Minecraft Expert Coding with Python

Scope and Sequence – Georgia

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 1: Variables Average Completion time: Approximately 5 hours	Learners will understand the definition of variables. Learners will understand the different types of variables and when they should be used in code. Learners will be able to create variables and assign them values. Learners will understand the definition of computing and computing networks.	**Pyth-3.2** Describe the relationship between variables and values when programming in Python. **Pyth-3.7** Identify and assign values and different data types to a variable in a program. **Pyth-3.8** Convert variable values between different data types in Python. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-7.3** Declare strings in programs using multiple declaration styles. **Pyth-7.4** Manipulate variables using string concatenation and slicing.	**3B-AP-12**: Compare and contrast fundamental data structures and their uses. **3A-AP-17:** Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. **3A-AP-21:** Evaluate and refine computational artifacts to make them more usable and accessible. **3B-AP-10**: Use and adapt classic algorithms to solve computational problems. **3B-CS-02:** Illustrate ways computing systems implement logic, input, and output through hardware components.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 2: Operators Average Completion time: Approximately 5 hours	Learners will understand operators and how they are used in code. Learners will be able to identify and use operators in Python. Learners will be able to evaluate expressions that use arithmetic operators. Learners will be able to write expressions using relational operators. Learners will be able to evaluate expressions that use relational operators.	**Pyth-3.3** Implement scripts that use logical, relational, boolean, and mathematical operators. **Pyth-3.6** Use arithmetic operators to modify variables in programs.	**3B-AP-12**: Compare and contrast fundamental data structures and their uses. **3B-CS-02:** Illustrate ways computing systems implement logic, input, and output through hardware components.
Lesson 3: Logic Average Completion time: Approximately 8 hours	Learners will understand what a conditional is and why it is used in code. Learners will learn what an If statement is and how to use it in their code. Learners will learn what an If / Else statement is and how to use it in their code.	**Pyth-3.4** Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements).	**3A-NI-04:** Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. **3B-AP-10:** Use and adapt classic algorithms to solve computational problems.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-2.2 Use various debugging and testing methods to ensure program correctness.	**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-17**: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. **3A-AP-21**: Evaluate and refine computational artifacts to make them more usable and accessible.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 4: Advanced Logic Average Completion time: Approximately 8 hours	Learners will learn what an If / Else If / Else statement is and why it is used in code. Learners will learn how to use an If / Else If / Else statement in their code. Learners will learn what nested conditionals are and why they are used in code. Learners will learn how to use nest conditionals in their code. Learners will learn what the different types of computing are including their strengths, weaknesses, and when to use them.	**Pyth-3.4** Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements). **Pyth-5.3** Implement conditional statements with the use of mathematical, relational, and Boolean operators. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop.	**3A-CS-02**: Compare levels of abstraction and interactions between application software, system software, and hardware layers **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-17:** Decompose problems into smaller components through systematic

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
			analysis, using constructs such as procedures, modules, and/or objects. **3A-AP-21:** Evaluate and refine computational artifacts to make them more usable and accessible. **3B-AP-10**: Use and adapt classic algorithms to solve computational problems.
Lesson 5: Loops Average Completion time: Approximately 8 hours	Learners will be able to identify and build code which uses while loop statements. Learners will be able to identify and build code which uses for loop statements. Learners will be able to identify and build code which uses nested loops.	**Pyth-5.4** Implement loops in programs using continue, break, and pass keywords. **Pyth-5.5** Describe the difference between for loops, while loops, for-each loops, infinite loops, and nested loops. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness.	**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-21:** Evaluate and refine computational artifacts to make them more usable and accessible. **3B-AP-10**: Use and adapt classic algorithms to solve computational problems.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 6: Coding Practice Average Completion time: Approximately 8 hours	Learners will be able to identify and build code that uses variables. Learners will be able to identify and use operators in code. Learners will be able to identify and build code using complex selections and conditionals. Learners will be able to identify and build loops and nested loops that use complex compound conditionals.	**Pyth-3.2** Describe the relationship between variables and values when programming in Python. **Pyth-3.7** Identify and assign values and different data types to a variable in a program. **Pyth-3.8** Convert variable values between different data types in Python. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-7.3** Declare strings in programs using multiple declaration styles. **Pyth-7.4** Manipulate variables using string concatenation and slicing. **Pyth-3.3** Implement scripts that use logical, relational, boolean, and mathematical operators. **Pyth-3.6** Use arithmetic operators to modify variables in programs.	**3B-AP-10:** Use and adapt classic algorithms to solve computational problems. **3B-AP-12**: Identify and Use Operators in Python **3A-AP-17**: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. **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.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-3.4 Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements). **Pyth-5.3** Implement conditional statements with the use of mathematical, relational, and Boolean operators. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop. **Pyth-5.4** Implement loops in programs using continue, break, and pass keywords. **Pyth-5.5** Describe the difference between for loops, while loops, for-each loops, infinite loops, and nested loops.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 7: Project Average Completion time: Approximately 10 hours	Learners will build a game using all the programming knowledge previously learned such as variables, operators, logic, and loops.	**Pyth-3.2** Describe the relationship between variables and values when programming in Python. **Pyth-3.7** Identify and assign values and different data types to a variable in a program. **Pyth-3.8** Convert variable values between different data types in Python. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-7.3** Declare strings in programs using multiple declaration styles. **Pyth-7.4** Manipulate variables using string concatenation and slicing.	3B-AP-12: Identify and Use Operators in Python 3A-AP-17: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 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.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-3.3 Implement scripts that use logical, relational, boolean, and mathematical operators. **Pyth-3.6** Use arithmetic operators to modify variables in programs. **Pyth-3.4** Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements). **Pyth-5.3** Implement conditional statements with the use of mathematical, relational, and Boolean operators. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop. **Pyth-5.4** Implement loops in programs using continue, break, and pass keywords.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-5.5 Describe the difference between for loops, while loops, for-each loops, infinite loops, and nested loops. **Pyth-1.2** Analyze a program and describe the structures of Python code. **Pyth-2.1** Design, develop, debug and implement computer programs.	
Lesson 8: Functions Average Completion time: Approximately 7 hours	Learners will be able to understand the concept of modularity and identify when modularity can or should be used in code Learners will be able to identify, understand, call, and build functions Learners will be able to identify, understand, call and build functions that take parameters or return data. Learners will be able to identify and understand algorithmic efficiency.	**Pyth-5.7** Describe how a function is implemented within a program. **Pyth-5.8** Implement different types of functions in a program including those with return statements and different parameters. **Pyth-9.3** Use procedures to reduce the complexity of a program. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness.	**3A-AP-17**: Decompose problems into smaller components through systematic analysis, using constructs 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. **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.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
			3A-AP-18: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. **3B-AP-14:** Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
Lesson 9: Algorithms Average Completion time: Approximately 7 hours	Learners will compare multiple algorithms to determine if they yield the same side effect or result. Learners will create algorithms. Learners will combine and modify existing algorithms.	**Pyth-9.1** Identify an algorithm in Python as a sequence of instructions that transform data or generate conclusions based on the data. **Pyth-9.2** Implement different algorithms in programs using Python including sorting, search and merge algorithms. **Pyth-9.3** Use procedures to reduce the complexity of a program.	**3B-AP-11:** Evaluate algorithms in terms of their efficiency, correctness, and clarity. **3A-AP-17:** Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects **3B-AP-10:** Use and adapt classic algorithms to solve computational problems.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-9.4 Identify an algorithm as being recursive. **Pyth-9.5** Implement recursion in search and sort algorithms. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-3.1** Describe how procedural programming is implemented using the Python programming language.	**3B-AP-22:** Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
Lesson 10: Collections Average Completion time: Approximately 7 hours	Learners will be able to identify, understand when to use and build code which uses lists (creation, indexing, modifying, traversing, common methods)	**Pyth-6.1** Identify a list as an ordered series of data under one variable name and accessed with numeric indices. **Pyth-6.2** Identify a data structure as a collection of data values and types of data structures in the Python programming language (list, tuple, dictionary and set).	**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.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-6.3 Determine which data structures are most appropriate to model the program data (list, tuple, dictionary, or set). **Pyth-6.4** Explain the operations that can be applied to data structures using Python including lists and dictionaries as objects that can be changed, and strings and tuples as object that cannot be changed. **Pyth-6.5** Implement lists, sets, dictionaries, and tuples as function parameters, return values and internal variables within function bodies. **Pyth-6.6** Differentiate between methods and functions and analyze the effect of a method call on a program. **Pyth-7.1** Identify strings as arrays of bytes representing Unicode characters. **Pyth-8.1** Explain the principles of object-oriented programming using the Python programming language as the use of data	**3A-AP-21**: Evaluate and refine computational artifacts to make them more usable and accessible.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		values in the form of fields and code in the form of procedures or methods. **Pyth-8.2** Describe the use of objects and recognize the difference between an object and an instance. **Pyth-8.3** Define objects in Python and use principles of object-oriented programming for declaring methods and combining classes. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-1.5** Create graphical organizers to model classes, attributes, methods and object relationships. **Pyth-5.9** Use external Python libraries to produce multi-media (visual or audio) outputs.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 11: Structure and Error Handling Average Completion time: Approximately 7 hours	Learners will be able to document code using comments and documentation stings. Learners will be able to correctly use indentation to structure code appropriately. Learners will describe the purpose of a code segment or program by writing documentation.	**Pyth-4.1** Create single-line comments in programs that explain the purpose of each function **Pyth-4.2** Create multi-line comments that highlight the precondition and post condition state of each function **Pyth-4.3** Students will understand the importance of and make use of self-documenting code **Pyth-1.4** Describe the process of compiling and running a program in Python. **Pyth-2.1** Design, develop, debug and implement computer programs. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness.	**3A-AP-21**: Evaluate and refine computational artifacts to make them more usable and accessible. **3A-AP-23**: Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. **3B-AP-20:** Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-2.3 Identify different types of errors that can occur including parameter mismatch and scope errors. **Pyth-3.9** Describe cryptography in Python as the use of libraries that provide encryption to secure data and information (i.e., cryptographic standard libraries).	
Lesson 12: Programming Practice Average Completion time: Approximately 5 hours	Learners will be able to identify and build code that uses variables. Learners will be able to identify and use operators in code. Learners will be able to identify and build code using complex selections and conditionals. Learners will be able to identify and build loops and nested loops that use complex compound conditionals. Learners will be able to identify, understand, call, and build functions	**Pyth-3.2** Describe the relationship between variables and values when programming in Python. **Pyth-3.7** Identify and assign values and different data types to a variable in a program. **Pyth-3.8** Convert variable values between different data types in Python. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-7.3** Declare strings in programs using multiple declaration styles.	**3B-AP-10:** Use and adapt classic algorithms to solve computational problems. **3B-AP-12**: Identify and Use Operators in Python **3A-AP-17**: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. **3A-AP-13**: Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
	Learners will create algorithms. Learners will combine and modify existing algorithms. Learners will be able to identify, understand when to use and build code which uses lists	**Pyth-7.4** Manipulate variables using string concatenation and slicing. **Pyth-3.3** Implement scripts that use logical, relational, boolean, and mathematical operators. **Pyth-3.6** Use arithmetic operators to modify variables in programs. **Pyth-3.4** Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements). **Pyth-5.3** Implement conditional statements with the use of mathematical, relational, and Boolean operators. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop.	**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. **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 of data and procedures, or independent but interrelated programs. **3B-AP-14**: Construct solutions to problems using student-created

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-5.4 Implement loops in programs using continue, break, and pass keywords. **Pyth-5.5** Describe the difference between for loops, while loops, for-each loops, infinite loops, and nested loops. **Pyth-5.7** Describe how a function is implemented within a program. **Pyth-5.8** Implement different types of functions in a program including those with return statements and different parameters. **Pyth-9.3** Use procedures to reduce the complexity of a program. **Pyth-9.1** Identify an algorithm in Python as a sequence of instructions that transform data or generate conclusions based on the data. **Pyth-9.2** Implement different algorithms in programs using Python including sorting, search and merge algorithms.	components, such as procedures, modules and/or objects. **3A-AP-21:** Evaluate and refine computational artifacts to make them more usable and accessible.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-9.4 Identify an algorithm as being recursive. **Pyth-9.5** Implement recursion in search and sort algorithms. **Pyth-3.1** Describe how procedural programming is implemented using the Python programming language. **Pyth-6.1** Identify a list as an ordered series of data under one variable name and accessed with numeric indices. **Pyth-6.2** Identify a data structure as a collection of data values and types of data structures in the Python programming language (list, tuple, dictionary and set). **Pyth-6.3** Determine which data structures are most appropriate to model the program data (list, tuple, dictionary, or set). **Pyth-6.4** Explain the operations that can be applied to data structures using Python including lists and dictionaries as objects	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		that can be changed, and strings and tuples as object that cannot be changed. **Pyth-6.5** Implement lists, sets, dictionaries, and tuples as function parameters, return values and internal variables within function bodies. **Pyth-6.6** Differentiate between methods and functions and analyze the effect of a method call on a program. **Pyth-7.1** Identify strings as arrays of bytes representing Unicode characters. **Pyth-8.1** Explain the principles of object-oriented programming using the Python programming language as the use of data values in the form of fields and code in the form of procedures or methods. **Pyth-8.2** Describe the use of objects and recognize the difference between an object and an instance.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-8.3 Define objects in Python and use principles of object-oriented programming for declaring methods and combining classes.	
Lesson 13: Planning a Product Average Completion time: Approximately 8 hours	Learners will design a program and its user interface. Learners will develop a program using a development process.	**Pyth-1** Use computational thinking and the Python programming language to solve problems and create programs for real world applications such as game development, data analysis, and the arts	**3B-AP-17:** Plan and develop programs for broad audiences using a software life cycle process. **3B-AP-21**: Develop and use a series of test cases to verify that a program performs according to its design specifications. **3B-AP-23:** Evaluate key qualities of a program through a process such as a code review. **3A-AP-19:** Systematically design and develop programs for broad audiences by incorporating feedback from users.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
Lesson 14: Building the Product Average Completion time: Approximately 15 hours	Learners will build a game using all the programming knowledge previously learned such as variables, operators, logic, loops, collections, algorithms, and functions.	**Pyth-3.2** Describe the relationship between variables and values when programming in Python. **Pyth-3.7** Identify and assign values and different data types to a variable in a program. **Pyth-3.8** Convert variable values between different data types in Python. **Pyth-2.2** Use various debugging and testing methods to ensure program correctness. **Pyth-7.3** Declare strings in programs using multiple declaration styles. **Pyth-7.4** Manipulate variables using string concatenation and slicing. **Pyth-3.3** Implement scripts that use logical, relational, boolean, and mathematical operators. **Pyth-3.6** Use arithmetic operators to modify variables in programs.	3B-AP-10: Use and adapt classic algorithms to solve computational problems. 3B-AP-12: Identify and Use Operators in Python 3A-AP-17: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 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. 3B-AP-15: Analyze a large-scale computational problem and identify

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-3.4 Describe and use different types of logical operators in Python. **Pyth-5.1** Analyze how control structures are used in programs using Python. **Pyth-5.2** Identify and describe the structure of different types of conditional statements (if statements). **Pyth-5.3** Implement conditional statements with the use of mathematical, relational, and Boolean operators. **Pyth-5.6** Analyze the effect of a variable in a conditional statement and a loop. **Pyth-5.4** Implement loops in programs using continue, break, and pass keywords. **Pyth-5.5** Describe the difference between for loops, while loops, for-each loops, infinite loops, and nested loops. **Pyth-5.7** Describe how a function is implemented within a program. **Pyth-5.8** Implement different types of functions in a program including those	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 of data and procedures, or independent but interrelated programs. **3B-AP-14:** Construct solutions to problems using student-created components, such as procedures, modules and/or objects. **3A-AP-21:** Evaluate and refine computational artifacts to make them more usable and accessible.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		with return statements and different parameters. **Pyth-9.3** Use procedures to reduce the complexity of a program. **Pyth-9.1** Identify an algorithm in Python as a sequence of instructions that transform data or generate conclusions based on the data. **Pyth-9.2** Implement different algorithms in programs using Python including sorting, search and merge algorithms. **Pyth-9.4** Identify an algorithm as being recursive. **Pyth-9.5** Implement recursion in search and sort algorithms. **Pyth-3.1** Describe how procedural programming is implemented using the Python programming language.	**3B-AP-17:** Plan and develop programs for broad audiences using a software life cycle process. **3B-AP-21:** Develop and use a series of test cases to verify that a program performs according to its design specifications. **3B-AP-23**: Evaluate key qualities of a program through a process such as a code review.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-6.1 Identify a list as an ordered series of data under one variable name and accessed with numeric indices. **Pyth-6.2** Identify a data structure as a collection of data values and types of data structures in the Python programming language (list, tuple, dictionary and set). **Pyth-6.3** Determine which data structures are most appropriate to model the program data (list, tuple, dictionary, or set). **Pyth-6.4** Explain the operations that can be applied to data structures using Python including lists and dictionaries as objects that can be changed, and strings and tuples as object that cannot be changed. **Pyth-6.5** Implement lists, sets, dictionaries, and tuples as function parameters, return values and internal variables within function bodies.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	Introduction to Python (11.08300)	CSTA Standards
		Pyth-6.6 Differentiate between methods and functions and analyze the effect of a method call on a program. **Pyth-7.1** Identify strings as arrays of bytes representing Unicode characters. **Pyth-8.1** Explain the principles of object-oriented programming using the Python programming language as the use of data values in the form of fields and code in the form of procedures or methods. **Pyth-8.2** Describe the use of objects and recognize the difference between an object and an instance. **Pyth-8.3** Define objects in Python and use principles of object-oriented programming for declaring methods and combining classes.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com