Chapter 1: Students should be able to:

- □ Evaluate compound expressions using rules of precedence and order of operations.
- □ Assign variables with allowed names and understand how to rebind that variable to new or different values.
- □ Utilize assigning multiple variables at the same time and understand when it might be useful to do so.
- □ Update variables using shorthand syntax. (`A += 1`)
- □ Define simple functions with inputs and outputs
- □ Import and use the `math` library for mathematical functions.
- □ Distinguish between and create Python objects of `int`, `float`, and `bool` types.
- □ Identify what operations are viable on different types of basic objects (eg. You can use `+` for floats, integers, and strings).
- □ Identify the resulting object type after an operation is performed (eg. Adding an `int` to a `float` results in a `float`).
- □ Convert between object types using built-in functions (e.g. `int` or `float`).
- □ Print variables or other text to the screen

Chapter 2: Students should be able to:

- □ Construct program flow controls through the use of `if`, `elif`, and `else` statements with appropriate syntax.
- □ Parse complicated `if, elif, else` conditionals to decide what the output of a script might be.
- □ Evaluate expressions utilizing the logical operators `or`, `and`, and `not`.
- □ Construct `while` loops with appropriate conditionals and which also terminate (no infinite loops!).
- □ Understand how nested loops behave and describe the output of each iteration of a set of nested loops.
- □ Understand what a predicate function is and be able to both understand and write one.
- □ Construct `for` loops with correct syntax over appropriate sequences.
- □ Identify situations where a `for` loop or a `while` loop might be more appropriate.
- □ Utilize the `range` function appropriately to construct ranges over desired intervals with valid step sizes.

Chapter 3: Students should be able to:

- □ Describe what an algorithm is
- □ Describe in plain English an algorithm for solving a basic task.
- □ Utilize the simple `english.py` library for certain word-related problems.
- □ Write a simple test function to test the correctness or output of another function.

Chapter 7: Students should be able to:

- □ Describe the difference between a number, and a representation of a number.
- □ Describe simple numbers in either decimal, binary, or hexadecimal representation.
- □ Describe why a computer's binary floating-point math sometimes gives slightly different results than our standard base-10 mathematical operations.
- □ Explain how Python represents characters internally as integers, according to an encoding scheme called Unicode, and how to convert back and forth between a character and its corresponding integer value.
- □ Define `str` objects and know what operations can (and can't) be done on strings.
- □ Access individual elements of a string through indexing.

□ Slice strings to extract desired pieces with a starting point, a stopping point, and a stride size.

□ Determine the number of characters in a string.

□ Iterate through the elements of a string.

□ Grow strings through concatenation.

□ Use built-in common string methods to manipulate or search strings.

□ Use `input` to get information from a user and understand what variable type is returned.

□ Create strings with variables embedded within them easily using f-strings