2: VARIABLES, EXPRESSIONS, AND STATEMENTS

2.1: VALUES AND TYPES
A value is one of the basic things a program works with, like a letter or a number. The values we have seen so far are 1, 2, and "Hello, World!"

2.2: VARIABLES
One of the most powerful features of a programming language is the ability to manipulate variables. A variable is a name that refers to a value. An assignment statement creates new variables and gives them values.

2.3: VARIABLE NAMES AND KEYWORDS
Programmers generally choose names for their variables that are meaningful and document what the variable is used for. Variable names can be arbitrarily long. They can contain both letters and numbers, but they cannot start with a number. It is legal to use uppercase letters, but it is a good idea to begin variable names with a lowercase letter.

2.4: STATEMENTS
A statement is a unit of code that the Python interpreter can execute. We have seen two kinds of statements: print being an expression statement and assignment.

2.5: OPERATORS AND OPERANDS
Operators are special symbols that represent computations like addition and multiplication. The values the operator is applied to are called operands.

2.6: EXPRESSIONS
An expression is a combination of values, variables, and operators. A value all by itself is considered an expression, and so is a variable, so the following are all legal expressions (assuming that the variable x has been assigned a value):

2.7: ORDER OF OPERATIONS
When more than one operator appears in an expression, the order of evaluation depends on the rules of precedence. For mathematical operators, Python follows mathematical convention.

2.8: MODULUS OPERATOR
The modulus operator works on integers and yields the remainder when the first operand is divided by the second. In Python, the modulus operator is a percent sign (%). The syntax is the same as for other operators.

2.9: STRING OPERATIONS
The + operator works with strings, but it is not addition in the mathematical sense. Instead it performs concatenation, which means joining the strings by linking them end to end.

2.10: ASKING THE USER FOR INPUT
Sometimes we would like to take the value for a variable from the user via their keyboard. Python provides a built-in function called input that gets input from the keyboard. When this function is called, the program stops and waits for the user to type something. When the user presses Return or Enter, the program resumes and input returns what the user typed as a string.

2.11: COMMENTS
As programs get bigger and more complicated, they get more difficult to read. Formal languages are dense, and it is often difficult to look at a piece of code and figure out what it is doing, or why. For this reason, it is a good idea to add notes to your programs to explain in natural language what the program is doing. These notes are called comments, and in Python they start with the # symbol.

2.12: CHOOSING MNEMONIC VARIABLE NAMES
As long as you follow the simple rules of variable naming, and avoid reserved words, you have a lot of choice when you name your variables. In the beginning, this choice can be confusing both when you read a program and when you write your own programs. For example, the following three programs are identical in terms of what they accomplish, but very different when you read them and try to understand them.

2.13: DEBUGGING
At this point, the syntax error you are most likely to make is an illegal variable name, like class and yield, which are keywords, or odd~job and US$, which contain illegal characters.

2.E: VARIABLES, EXPRESSIONS, AND STATEMENTS (EXERCISES)
2.G: GLOSSARY