**Python for Everyone: Notes**

**Chapter 2: Variables, Expressions, and Statements**

* Constants
  + Fixed values such as numbers, letters, and strings, are called “constants” because their value does not change
  + Numeric constants are as expected
  + String constants use single or double quotes
* Variables
  + A variable is a named place in the memory where a programmer can store data and later retrieve the data using the variable “name”
  + Programmers get to choose the names of the variables
  + You can change the contents of a variable in a later statement
* Python variable name rules
  + Must start with a letter or underscore
  + Must consist of letters, numbers, and underscores
  + Case sensitive
* Mnemonic variable names
  + Since programmers are given a choice in how we choose our variable names, there is a bit of best practice
  + We name variables to help us remember what we intend to store in them (mnemonic = memory aid)
  + This can confuse beginning students because well named variables often sound so good that they must be keywords
* Sentences or lines
  + X = 2 <- assignment statement
  + X = x + 2 <- assignment with expression
  + Print(x) <- print statement
* Assignment statements
  + We assign a value to a variable using the assignment statement (=)
  + An assignment statement consist of an expression on the right-hand side and a variable to store the results
    - X = 3.9 \* x \* ( 1 – x)
  + A variable is a memory location used to store a value
  + The right side is an expression
  + Once the expression is evaluated, the result is assigned to x
  + A variable is a memory location used to store a value. The value stored in a variable can be updated by replacing the old value to a new value
  + The right side is an expression, once the expression is evaluated, the result is placed in the variable on the left side
* Numeric expressions
  + Because of the lack of mathematical symbols on computer keyboards, we use computer speak to express the classic math operations
  + Aseterick is multiplication
  + Expotentiation looks different than in math
  + Operatons / Operations: + addition, - subtraction, \* multiplication, / division, \*\* power, % remainder
* Order of evaluation
  + When we string operators together, python must know which one to do first
  + This is called operation precedence
* Operator precedence rules
  + Highest precedence rule to lowest precedence rule
    - Parentheses are always respected
    - Exponentiation (power)
    - Multiplication, division, remainder
    - Addition, subtraction
    - Left right
* What does type mean
  + In python, variables, literals, and constants have a type
  + Python knows the difference between an integer number and a string
  + For example, + means addition, if something is a number and concatenate if something is a string
  + Concatenate = put together
* Type matters
  + Python knows what type everything is
  + Some operations are prohibited
  + You cannot add 1 to a string
  + We can ask python what type something is by using the type() function
* Several types of numbers
  + Numbers have two main type
    - Integers are whole numbers:
      * -14, -2, 0, 1, 100
    - Floating points have decimal parts:
      * -2.5, 0.0, 98.6, 14.0
  + There are other number types, they are variations on float and integer
* Type conversion
  + When you put an integer and floating point in an expression, the integer is implicitly converted to a float
  + You can control this with the built-in fuction int() and float()
* Integer division:
  + Integer division produces a floating point result, different than from python 2.x
* String conversion
  + You can also use int() and float() to convert between strings and integers
  + You will get an error if the string does not contain numer characters
* User input
  + We can instruct python to pause and read ata from the user using the input() function
  + The input() function returns a string
* Converting user input
  + If we want to read a number from the user, we must convert it from a string to a number using a type conversion function
* Comments in python
  + Anything after # is ignored by python
  + Why comment
    - Describe what is going to happen in a sequence of code
    - Documents who wrote the code or other ancillary information
    - Turn off a line of code perhaps temporarily