Intro to Python: Class 3 Outline

Note: Concepts introduced here will be demonstrated using sample scripts in: https://github.com/skoshiwosh/democode

1) Control Flow
   - Demo these using demoscript.py example: if, try, for, while
   - The if statement is used to check a condition: *if* the condition is true, we run a block of statements (called the *if-block*), *else* we process another block of statements (called the *else-block*). The *else* clause is optional.
   - An if/elif-statement is a generalized if-statement with more than one condition. It is used for making complex decisions.
   - The while statement allows you to repeatedly execute a block of statements as long as a condition is true. A while statement is an example of what is called a *looping* statement. A while statement can have an optional else clause.
   - The `for..in` statement is another looping statement which *iterates* over a sequence of objects.
   - The `break` statement is used to *break* out of a loop statement i.e. stop the execution of a looping statement, even if the loop condition has not become `False` or the sequence of items has not been completely iterated over.
   - The `continue` statement is used to tell Python to skip the rest of the statements in the current loop block and to *continue* to the next iteration of the loop.
   - Use the `try..except` statement to handle exceptions, that is to execute alternative block of code if code under try causes an error.

2) Functions, Modules and Packages
   - Demo builtin utility functions: type, dir, id, len, iter, next, eval, help,
   - Demo builtin type converters: int, float, str, list, tuple, set, dict, bin, hex, chr
   - Demo builtin math functions: abs, cmp, divmod, min, max, round, sum
   - Review standard modules used often: sys, os, glob, string, math, logging, etc
   - Demo custom functions: definition, argument

3) Dictionaries
   - Introduce definition and use of dictionaries

4) Scope of Variables and Script Structure
   - When you declare variables inside a function definition, they are not related in any way to other variables with the same names used outside the function - i.e. variable names are *local* to the function. This is called the *scope* of the variable. All variables have the scope of the block they are declared in starting from the point of definition of the name.