Refreshed knowledge on Python using this video (length: ~4½ hrs) and random google searches. Current time: 2:00:37.

NOTES:

Using PyCharm as I have familiarity with JetBrains products, would like to avoid using a text editor like VIM, and want to be able to easily follow along with the video (where they use this application).

The video covers very simple stuff, so note taking and application development will probably not be as active during ~the first half of the video.

Variables are created without type declaration… Strange.

Comments use #

String concatenation does not automatically parse integers into strings. Python also does not seem to use the standard toString() function (instead you cast the variable using str()).

Python still uses escape sequences.

If statements are formatted as if statement: else: (and indentation indicates what belongs to what in an if statement)

Index is used to provide the location of a character or sequence of characters

Strings appear to automatically work as arrays (arrays of chars, presumably)

The .replace() function is cool–can replace words or even parts of words in a String.

The round() function is cool in that it chooses between ceil and floor on its own.

lower() and upper() are the replacements for toLower() and toUpper().

input() is the user input device and can be used in a simple assignment statement.

Lists can hold different types (eg. Strings and integers and booleans and etc)

Lists index negative numbers can be used to go from the opposite end of the list (eg. -1 would be the last element). This only works through one loop though (eg. can't do -4 on a 4 element list expecting to get -1 again)

List[N:] gets element N forward and List[:N] gets every element before N. Obviously List[N:M] works too (grabs elements N, N+1, N+2…. M-1)

List.extend(List2) appends list2 to the end of list

List.append(x) adds x to list

List.insert(index, value)

List.clear()

List.index(x)

List.count(x)

List.sort() ← this is cool, alphabetical/ascending

List.copy()

Tuples can't change individually, but can change if you reassign the whole thing (eg. myval = (1) instead of myval[0] = 1)

Declare function by using def

or / and statements are done using simply or / and

not is simply not

Else if is elif

Comparisons seem to be mostly the same (<,>,<=,>=, !=, ==)