Week6, Lecture2

Today's topic continues from where we left off on Monday. We are talking about parameter passing (i.e., passing variables to functions) in such a way that a called function B() can work on and change values in the calling function A().

We start by looking at how variables are assigned in Python, and then recall that (a) parameters and other variables in a function are local to the function, and (b) Python passes parameters by value, meaning that if you pass an object as a parameter, the function gets to work with a copy of the object.

So when you pass a list L [777, 888] to a function, list L[] has two entries: the first points to int object 777, and the second points to int object 888. Inside this same function, if the entries L[0] and L[1] are updated, now list L[] will look different because L[0] and L[1] will point to the new updated objects, and the old objects 777 and 888 will be "lost" to the program, and collected by the garbage collector.

The first 3 examples from today, and 5.py and 6.py from the Monday lecture covers what you need to know. The rest will come with practice, as you write functions to do this for your own applications.

Examples 4.py and 5.py from today's lecture deal with

“sequence encoding”. The files explain what is going on in each example.

I may add video on this later in the week if I can.

You should be comfortable with parameter passing in functions, sequence encoding (“ciphers’, such as done in this simple example with a dictionary), and while/for loops with nesting. Nested loops are common, especially 2, 3 and 4 levels of nesting.