A screenshot of a video game

Description automatically generated

In this class we are learning id() function, type() function, mutable and immutable objects, and arguments passed in functions. This knowledge will give the efficacy each program runs and why it is either true, false or what the outputs are.

Id() returns a certain id that is unique for an object. Every object has its own unique id; almost like a pid in C. To use it you could enter print(id(a)). It is the memory address found each time a program is ran. Type() can output the type of whatever the variable may be. If we are trying to find the type of variable a, we would use print(type(a)). You could even use it as a condition to handle errors if a tuple or dictionary are not as such, create it and give out an error type.

Mutable objects can be changed, lists are mutable when using [] ; on the left side of a dictation that list will change depending on the new variable. Also with append();

tree = [1, 2]

tree.append(3)

output:

[1, 2, 3]

we can change elements in a given list and output the new list. We can even omit elements in a list using pop(), using del function and clarifying which element(s) to be deleted.

Not all things are mutable, i.e. strings. Once you enter a string it cannot be changed.

Motto = [“braging rights”]

Motto.append[4] = “g”

Will output an error

Even data types, like integer, tuples, float, character, bools cannot be changed after they are created. To do so, you’d only find error codes in your path.

When coding it is vital to choose mutable objects and immutable objects depending on what the program is intended to do. When choosing mutable, it gives the program directions to change as the client or company needs it to. Making the system work for the clients changing needs successfully. Immutable objects do not take the program long to run and cuts down on run time because the values never change because they are stored at their specific memory address. In the case of user input run time would be beneficial here. Immutable objects also make the program easier to debug since they cannot be modified after created.

Arguments can be called by reference or value and id can help find this address as well. If the program needed to determine and test whether or not a certain object is at a specific location, using key words like is or is not can determine if the object is what the program needs. So in order to call on it successfully, the value would be entered in () after the function. Because if not a mutable objects value can be changed, whereas an immutable objects does not.