Mod-2-3-Notes

**Decorators**

* Decorators allows programmers to modify the behavior of a function or class. They **wrap another function in order to extend the behavior of the wrapped function without permanently modifying it.**
* Key Elements To Note:
  + In Python, functions are **first class objects** which means the functions in Python can be **used or passed as arguments.**
    - A function is **an instance of the Object type**
    - You can **store the function in a variable**
    - You can **pass the function as a parameter to another function**
    - You can **store them in data structures** such as hash tables, lists, etc.
* A **function decorator:** a function that takes in another function as an argument.

The image below is my Colab code submission. The point of using a decorator method is to pass another method with arguments or without to it in order to perform methods on the arguments.

* The modifications are done within the wrapper method defined inside the decorator method
* The method that has been decorated can return a value and be used if need be. In order to use that returned value you would simply need to invoke the function parameter func() within the wrapper method defined in the decorator. For example inside wrapper, if the decorated method meatball were to return a number you would acquire it inside wrapper using func().
* The decorator returns the wrapper method. This is simply so that the wrapper function is invoked.

A computer screen shot of text

Description automatically generated