Lesson 11 Notes OOP Architecture in Python

* Encapsulation is the purpose of Object Oriented Programing and it refers to the packing of data members(primitive data, methods, ect.) into and object and restricting outside access to them (information hiding)
* Constructors are like methods but they only have one pre-defined purpose and its always called “\_\_init\_\_”, followed by parameters
* All instance variables must have the “self.” prefix. self refers to the object being created, which the instance variables belong to
* Once objects are created we can use their functions to provide controlled access to data
* Using modifier functions, we can modify our private instance data from outside the object under a specific set of rules
* To retrieve instance variable data from our objects we can use accessors, which are a special kind of function that returns instance-level data without giving users the ability to modify the data
* The main() function is your entry point into any object oriented program
* **class OOP\_Structure**:
* #Constructor
* **def** \_\_init\_\_(self, fName="", lName="", uName=""):
* self.firsName = fName
* self.lastName = lName
* self.userName = uName
* #Modifier
* **def setUName**(self, newUser):
* self.userName = newUser
* #Accessors
* **def getFirstName**(self):
* **return** self.firsName
* **def getLastName**(self):
* **return** self.lastName
* **def getUserName**(self):
* **return** self.userName
* **def main**():
* firstName = input("Enter your first name: ")
* lastName = input("Enter your last name: ")
* userName = input("Enter your desired user name: ")
* user1 = OOP\_Structure(firstName, lastName, userName)
* print("<<<<<<<<USER INFO>>>>>>>>>>>>>")
* print("Name: ", user1.getFirstName(), " ", user1.getLastName())
* print("User Name: ", user1.getUserName(), "\n\n")
* user1.setUName("pHandsome")
* print("<<<<<<<<USER INFO>>>>>>>>>>>>>")
* print("Name: ", user1.getFirstName(), " ", user1.getLastName())
* print("User Name: ", user1.getUserName(), "\n\n")
* Creating an object is referred to as creating an instance of class or “instantiating” an object
* The constructor only runs once, during object instantiation (creating the object)
* The constructor is not part of the object once it is instantiated (created), and we cannot call it again. This is why we need modifiers (“setters”) – to change the data values on our objects.
* Encapsulation means data hiding
* The constructor only runs once, during object instantiation and is not part of the object once it is instantiated and we cannot call it again and this is why we need modifiers