Python Advance Assignment_2

Name : Bhavik Modi

E-mail : bmodi700@gmail.com

Website : https://keenbm.github.io/

Q1. What is the relationship between classes and modules?

A Python class is like an outline/blueprint/mold for creating a new object. An object is anything that you wish to manipulate or change while working through the code. Every time a class object is instantiated, which is when we declare a variable, a new object is initiated from scratch

Whereas in Python, Modules are simply files with the . py extension containing Python code that can be imported inside another Python Program. In simple terms, we can consider a module to be the same as a code library or a file that contains a set of functions/Classes that you want to include in your application.

Q2. How do you make instances and classes?

For creating a class instance. we call a class by its name and pass the arguments which its init method accepts.

Example: exampleObj = SampleClass('Salary',20000), Here exampleObj is an instance of class SampleClass with attriubutes 'Salary' and 20000.

Whereas for creating a class, we use the Class keyword. Class keyword is followed by classname and semicolon.

Q3. Where and how should be class attributes created?

In [1]:

```
#Here Salary is a class created with class keyword with arguments salary
and amount.

class Employee:
    def __init__(self, salary,amount):
        self.salary = salary
        self.amount = amount
```

Q4. Where and how are instance attributes created?

Instances attributes are passed to the class when an object of the class is created. Unlike class attributes, instance attributes are not shared by all objects of the classs. instead each object maintains its own copy of instance attributes at object level. whereas incase of class attributes all instances of class refer to a single copy. Usually instance attributes are defined within the **init** method of class

Q5. What does the term "self" in a Python class mean?

self represents the instance of the class (it represents the object itself). By using the “self” keyword we can access the attributes and methods of the class with in the class in python. It binds the attributes with the given arguments.

Q6. How does a Python class handle operator overloading?

Python Classes handle operator overloading by using special methods called Magic methods. these special methods usually begin and end with __ (double underscore) Example: Magic methods for basic arithmetic operators are:

- -> **add**()
- -> **sub**()
- -> **mul**() / -> **div**()

Q7. When do you consider allowing operator overloading of your classes?

When we want to have different meaning for the same operator accroding to the context we use operator overloading.

Q8. What is the most popular form of operator overloading?

The most popular form of operator overloading in python is by special methods called Magic methods. Which usually beign and end with double underscore .

Q9. What are the two most important concepts to grasp in order to comprehend Python OOP code?

Classes and objects are the two concepts to comprehend python OOP code as more formally objects are entities that represent instances of general abstract concept called class

Along with classes and objects the important concepts to grasp are:

Inheritence Abstraction Polymorphism Encapsulation