Object oriented programming with python:

Learning Python classes and objects, Built-In Class Attributes and Class Methods

We know that,

- Python is a scripting language
- Python is functional language
- Python is modular programming language

In python everything is object. If we write, l = [1,2,3,4,5], this is list object. With the help of this object, we can call method like l.append(50). Important thins is list is already a pre-defined class and we can create object of that.

Now, our requirement is to create object of our own class. Now, understand meaning of object, class, reference variable.

```
Plan===>We can construct any number of buildings
plan acts as blueprint to construct building
plan==>class
Building===>object
```

```
class is a blue print to construct objects

object is physical existence of a class/Real world entity
```

Consider the example of bank customer, it include all the properties of that bank customers and all the operations which customer can perform.

Here, customer is a class and properties + operations/methods is a part of class.

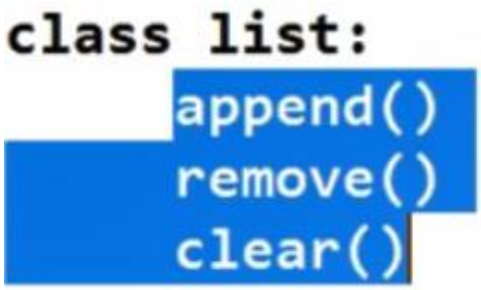
```
customer class
   properties+operations/methods
```

Class

The class can be defined as a collection of objects. It is a logical entity that has some specific attributes and methods. For example: if you have an employee class then it should contain an attribute and method, i.e. an email id, name, age, salary, etc.

Object

The object is an entity that has state and behavior. It may be any real-world object like the mouse, keyboard, chair, table, pen, etc. Everything in Python is an object, and almost everything has attributes and methods. Object is a physical existing




```
l=[10,20,30,40]===>list object
l.append(10)
l.remove(10)
```

With the help of l , we can access properties of list class. So, l is a reference variable.

```
class==>blue print
object==>physical existence of a class
reference variable==>which is pointing to our object
by using that we can perform operations on that object
```

So, to create object, blueprint (class) must be required.

Class is a combinations of properties (variables) and actions (methods).

How to define a class:

We can define class with class keyword.

class class_name:

variables (properties)

methods (behaviour / action)

class students:

name, age, marks, roll_num

read()

write()

sleep()

Consider the class demo, where documentation string is there. The documentation string is represented can be represented between ‘’’ ’’’.

Class classname:

‘’’ doc string content ’’’

Variables (properties)

Methods (actions / behaviour)

Class student:

‘’’ This is demo example ’’’

Name
Age
Marks
read()
write()
sleep()

These things as a part of this class is called as blueprint.

To access document string:

Print(className.__doc__)

In python how many types of variables are there?

I am not talking about variable as in case of function programming (i.e., local and global). But in case of object oriented programming, we have

1. Instance Variables
2. Static Variables
3. Local Variables

The way in case of functional programming, there are private method and public method. In case of python, methods can be of types:

1. Instance Methods
2. Class Methods ➔ @classmethod
3. Static Methods ➔ @staticmethhod

Now, consider the example given below:

```
class student:
    def __init__(self):
        self.name = "Karan"
        self.age = 27
        self.marks = 80
    def display(self):
        print("Hello... My name is : "+self.name)
        print("My Age is : "+self.age)
        print("My percentage in exam is : "+self.marks)
```

Here, __init__ can be treated as the way constructor works. Purpose of that __init__ method is to initialize variables.

Now, consider the purpose of self. We know that

If we take simple function like:

def demo():

This function we can access without using object. Just by using function name is enough.

But, in case of method which is part of class, we must require to create object of that class. And for object there must be some variable pointing to that object called reference variable.

The variable pointing to current object is self.

self is implicit variable (reference variable), pointing to object.

So, we can say, current object name is Karan, current object age is 27 like wise.

Now, lets create object.

In JAVA,

Student s = new Student();

In case of python:

s = student()

s.display()

Here, the variables are initialized as a hardcoded method. Values are fixed. Now, look at this dynamic way of creating and initializing class variables.

Note:

Refer Inheritance, Abstraction, Polymorphism concepts from
https://www.javatpoint.com/inheritance-in-python

https://www.javatpoint.com/python-constructors

Also refer practical from attached .rar file.