Day15_Class and Object

Python is an Object oriented programming since it is developed on the two basic fundamental concepts known as a class and an object .

Class : A class is a logical entity(person, place or a thing)
Ex: Employee
A class contain variable(data members) and functions(methods)
Ex: Employee has eno, ename ,esal as the data members
Employee can have the functionalities such as displayDetails()
Object : An Object is a physical entity which is an instance of the class
Ex: Jack is an Employee
An Object help us to access the data members and the functions(methods)

In python we have two types of classes
1. **Predefined classes**
2. **User-defined classes**

Predefined classes : The classes that are already created and as a programmer we create the Objects to access the variables and methods
Ex: str,list,…

```
var1 = 40
print(type(var1))
#<class 'int'>

lst=[15,12,18]
print(type(lst))

#<class 'list'>
```

User-defined classes : These are the type of classes that are defined by the user by having his own variables and methods

```
#Syntax:
class classname:
   class variables
   class methods
   instance methods
      instance variables
  constructors
```

#Example
Class variables are the variables declared inside the class and outside the methods and are accessible using the class instance variable and the classname also inside and outside the class

```
class Employee:
   eno = 5614
   ename= "Jack"
   esal = 50000


e1 = Employee()
#e1-->class instance variable
#Employee() --> instantiating the object
print(e1.eno)
print(e1.ename)
print(e1.esal)
5614
Jack
50000

print(Employee.eno)
print(Employee.ename)
```

```
print(Employee.esal)
5614
Jack
50000
```

class methods : The methods(functions) created inside the class and are accessed or invoked using the class name only

```
class Employee:
   eno = 5614
   ename= "Jack"
   esal = 50000
   def display():
      print(Employee.eno)
      print(Employee.ename)
      print(Employee.esal)

Employee.display()
5614
Jack
50000
```

Instance methods : The methods defined by using 'self' keyword as the first parameter inside the method indicating that this method will be accessed by using the instance. This method may contain instance variables. There is no need to pass any value onto the self parameter

Instance variables : The variables declared inside the instance methods using self.variable name and are accessed through out the class

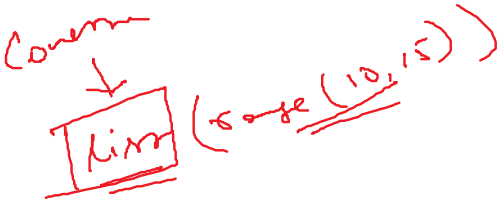
```
class Employee:
   def setDetails(self):
      self.eno = 5614
      self.ename= "Jack"
      self.esal = 50000
   def display(self):
      print(self.eno)
      print(self.ename)
      print(self.esal)


Employee.setDetails()
#TypeError: setDetails() missing 1 required positional argument: 'self'

e1=Employee()
e1.setDetails()

e1.display()

class Employee:
   def setDetails(self,e,n,s):
      self.eno = e
      self.ename= n
      self.esal = s
   def display(self):
      print(self.eno)
      print(self.ename)
      print(self.esal)

e1 = Employee()
e1.setDetails(1011,"Peter",45000)
e1.display()
1011
Peter
```

45000

Constructor :

1. A constructor is a method that is defined using __init__(self,par1,par2,…) and can be called or invoked by using the class name().

2. It is mostly used for initializing the variables that required for all the methods



```
class Employee:
   def __init__(self,e,n,s):
      self.eno = e
      self.ename= n
      self.esal = s
   def display(self):
      print(self.eno)
      print(self.ename)
      print(self.esal)

e1 = Employee(10,"Monie",55000)
e1.display()
10
Monie
55000
```