Day 7

Date 13 June 2024

Daily Report

Today's training was based on Object Oriented Programming in Python and one main concept of Object Oriented Programming - classes & Objects.

Today's Topic Covered

Object Oriented Concept

This approach allows us to club together the data and behaviour so that it become easier to code real world scenarios.

Class & Object

Objects are real world entities and classes are blue print of objects.A class is a classification of certain objects and it is just a descreption of the properites and behaviour of ojects.

```
class class_name:


object_name = class_name(attributes)
```

Constructor

using --init-- function is used to add attributes in class. This function is invoke automatic when object is created.This function is called a constructor.

self is keyword used as a permanent argument in classes to access any attribute.

```
class class_name:
  def __init__(self,arguments):
    self.variable = argument
  def class_function(self,argument):
    #code

object = class_name(arguemnts)
object.argument
object.class_fuction(argument)
```

some practice Question:-

```
class car:
  def __init__(self,make,model,year):
    self.make = make
    self.model = model
    self.year = year
mod1 = car("hyundai",'ABC','1989')
print("Company : ",mod1.make)
print("Model: ",mod1.model)
print("Year: ",mod1.year)
```

```
Company :  hyundai
Model:  ABC
Year:  1989
```

```
class person:
  def __init__(self,name,age):
    self.name = name
    self.age = age
  def greeting(self):
    print("Hello, my name is ",self.name," .I am ",self.age," years old.")
per1 = person("John",25)
per1.greeting()
```

```
Hello, my name is  John  .I am  25  years old.
```

```
class rectangle:
  def __init__(self,length,width):
    self.l = length
    self.w = width
  def Area(self):
    print(self.l * self.w)
rec = rectangle(20,10)
rec.Area()
```

```
200
```

```
class student:
  def __init__(self,name,grade):
    self.name = name
    self.grade = grade
  def average_grade(self):
    print("average grade : ",sum(self.grade)/len(self.grade))
num = [56,89,67,90]
st = student("John",num)
print("John: ")
st.average_grade()
```

```
John:
average grade :  75.5
```

```
class book:
  def __init__(self,title,author,pages):
    self.title = title
    self.author = author
    self.pages = pages
  def display(self):
    print("Title : ",self.title.title(),"\nAuthor: ",self.author,"\nPages : ",self.pages)
b1 = book("Nature","John legend",289)
b1.display()
```

```
Title :  Nature
Author:  John legend
Pages :  289
```

```
class dog:
  def __init__(self,name,breed):
    self.name = name
    self.breed = breed
  def bark(self):
    print("Bark!!")
d1 = dog("Tiger","ABC")
d1.bark()
```

```
Bark!!
```

```
class BankAccount:
  def __init__(self,account_no,balance):
    self.acc = account_no
    self.bal = balance

  def deposit(self,dep):
    self.bal = self.bal + dep

  def withdaw(self,w):
    self.bal = self.bal - w

  def display(self):
    print("Balance : ",self.bal)

bank = BankAccount(123456,20000)
bank.deposit(10000)
bank.display()
bank.withdaw(5000)
bank.display()
```

```
Balance :  30000
Balance :  25000
```

```
class laptop:
  def __init__(self,brand,price):
    self.brd = brand
    self.pr = price

  def discount(self):
    dis = 10
    price = self.pr - self.pr*10/100
    print(price)
mob1 = laptop("Apple",50000)
mob1.discount()
```

```
45000.0
```

```
class employee:
  def __init__(self,name,salary):
    self.name = name
    self.sal = salary
  def display(self,rais):
    self.sal += rais
  def dis(self):
    print("Salary : ",self.sal)
em1 = employee("John",40000)
em1.dis()
em1.display(5600)
em1.dis()
```

```
Salary :  40000
Salary :  45600
```

```
class point:
  def __init__(self,x,y):
    self.x = x
    self.y = y

  def distance(self,q_point):
    distance = ((self.x - q_point.x)**2 + (self.y - q_point.y)**2)**0.5
    return distance

p = point(4,5)
q = point(6,7)
p.distance(q)
```

```
2.8284271247461903
```

```
class movie:
  def __init__(self,name,director,year):
    self.name = name
    self.director = director
    self.year = year

  def display(self):
    print("Movie Name : ",self.name,"\nDirector : ",self.director,"\nRelease Year : ",self.y

m1 = movie("Avengers","Peter",2012)
m1.display()
```

```
Movie Name :  Avengers
Director :  Peter
Release Year :  2012
```

```
class product:
  def __init__(self,name,price):
    self.name = name
    self.price = price

  def final_price(self):
    tax = 2
    f_price = self.price + self.price * tax/100
    return f_price

p = product("rice",45)
p.final_price()
```

```
45.9
```

```
class player:
  def __init__(self,name,score):
    self.name = name
    self.score = score

  def update(self,new_score):
    self.score += new_score
    return self.score

p = player("John",78)
p.update(10)
```

```
88
```

```
class House:
  def __init__(self,address,no_of_room):
    self.address = address
    self.no = no_of_room

  def display(self):
    print("Address: ",self.address,"\nNumber of Room: ",self.no)

h = House("#123 house no,21 street ABC",5)
h.display()
```

```
Address:  #123 house no,21 street ABC
Number of Room:  5
```

```
class shape:
  def __init__(self,color,filled):
    self.color = color
    self.filled = filled

  def properties(self):
      print("color: ",self.color,"\nFilled : ",self.filled)

s = shape("red",True)
s.properties()
```

```
color:  red
Filled :  True
```

```
class person:
  def __init__(self,name,age):
    self.name = name
    self.age  = age
  def compare(self,sec):
    if(self.age >sec.age):
      print(f"{sec.name} is younger than {self.name}.")
    elif(self.age<sec.age):
      print(f"{sec.name} is older than {self.name}")
    else:
      print(f"{sec.name} is same age as {self.name}")

p1 = person("John",25)
p2 = person("Peter",35)
p3 = person("Marco",25)
p4 = person("Harry",17)
p1.compare(p2)
p1.compare(p3)
p1.compare(p4)
```

```
Peter is older than John
Marco is same age as John
Harry is younger than John.
```

```
class calculator:
  def __init__(self,x,y):
    self.x = x
    self.y = y
  def add(self):
    return self.x+self.y
  def subtract(self):
    return self.x - self.y
  def multiply(self):
    return self.x*self.y
  def divide(self):
    return self.x/self.y

p = calculator(20,4)
print("Addition: ",p.add())
print("Subtraction: ",p.subtract())
print("Multiply: ",p.multiply())
print("Divide: ",p.divide())
```

```
Addition:  24
Subtraction:  16
Multiply:  80
Divide:  5.0
```

```
class number:
  def __init__(self,num):
    self.num = num
  def ones(self):
    if(self.num > 0):
      return self.num
    else:
      return 0
  def threes(self):
    if(self.num>0):
      return self.num//3
    else:
      return 0
class user:
  c = 0
  def __init__(self):
    pass
    user.c +=1
u1 = user()
u2 = user()
u3 = user()
print(user.c)
```

```
3
```

```
n2 = number(45)
print("Number: 45")
n2.check()
```

```
Number = 5
  ones:  5
  threes:  1
  Nine:  0
Number: 45
  ones:  45
  threes:  15
  Nine:  5
```