**13 June 2024**

TOPICS: **OPPS CONCEPT CLASSES & OBJECTS, ATTRIBUTES, VARIABLE VS ATTRIBUTES, CREATING ATTRIBUTES IN A CLASS, CONSTRUCTOR(TYPES)**

1.Define a simple Car class with attributes for make, model, and year. Create an object of this class and print its attributes.

```
class Car:
  pass

car1=Car()
car2=Car()

car1.make="Hundyai"
car1.model="Verna"
car1.year="1988"

car2.make="abc"
car2.model="Swift"
car2.year="2006"

print(car1.make)
print(car2.model)
```

```
Hundyai
Swift
```

2.Create a Person class with attributes for name and age. Add a method to print a greeting message. Create an object and call the method.

```
class Person:
  def __init__(self,name,age):
    self.name=name
    self.age=age

p1=Person("jashandeep",19)
p2=Person("komalpreet",20)

print("First Person's name" ,p1.name, "and her age is ",p1.age)
print("Second Person's name" , p2.name, "and her age is ",p2.age)
```

```
First Person's name jashandeep and her age is  19
Second Person's name komalpreet and her age is  20
```

3.Define a Rectangle class with attributes for length and width. Add a method to calculate the area of the rectangle. Create an object and print the area.

```
class Rectangle:
  def __init__(self, length,width):
    self.length=length
    self.width=width
  def calculate_area(self):
    value=self.length
    value1=self.width
    return value*value1
R=Rectangle(3,7)
R.calculate_area()
```

```
21
```

4.Create a Student class with attributes for name and grades (a list of numbers). Add a method to calculate the average grade. Create an object and print the average grade.

```
class Student :
 def __init__(self,name,marks):
    self.name=name
    self.marks=marks

 def avg(self):
    sum=0
    for val in self.marks:
      sum += val

    print(self.name,"your avg score",sum/3)

s1=Student("karan",[77,78,89])
s1.avg()
```

```
karan your avg score 81.33333333333333
```

5.Define a Book class with attributes for title, author, and pages. Add a method to display the book's details. Create an object and call the method.

```
class Book:
  def __init__(self,title,author,pages):
    self.title=title
    self.author=author
    self.pages=pages

book1=Book("THE GREAT INDIANS","ABC","278")
book2=Book("THE GAI","IK","345")

print("our first book title",book1.title," ,name of the author",book1.author," and number of pages",book1.pages)
print("our first book title",book2.title," ,name of the author",book2.author," and number of pages",book2.pages)
```

```
our first book title THE GREAT INDIANS  ,name of the author ABC  and number of pages 278
our first book title THE GAI  ,name of the author IK  and number of pages 345
```

6.Create a Dog class with attributes for name and breed. Add a method to make the dog bark (print a bark message). Create an object and call the method

```
class Dog:
  def __init__(self,name,breed):
    self.name=name
    self.breed=breed
  def action(self):
    print("Bark")
print("Dog-1")
d1=Dog("abc","Germanshffd")
d1.action()
```

```
Dog-1
Bark
```

7.Define a BankAccount class with attributes for account number and balance. Add methods to deposit and withdraw money. Create an object and test the methods.

```
class BankAccount:
  def __init__(self,account_no,balance):
    self.account=account_no
    self.balance=balance
  def deposite(self,money):
    if money>0:
      self.balance+=money
      return True
    else:
      return False
  def withdraw(self,money):
    if 0<money<=self.balance:
      self.balance-=money
      return True
    else:
      return False
  def get_balance(self):
    return self.balance
Amount=BankAccount("12345678",600000)
Amount.deposite(40000)
Amount.withdraw(100000)
Amount.get_balance()
```

```
540000
```

8.Create a Circle class with attributes for radius. Add a method to calculate the circumference. Create an object and print the circumference

```
class Circle:
  def __init__(self,radius):
    self.radius=radius
  def calculate_circumference(self):
    r2=self.radius
    return 2*3.14*r2
r=Circle(4)
r.calculate_circumference()
```

```
25.12
```

9.Define a Laptop class with attributes for brand and price. Add a method to apply a discount to the price. Create an object and test the method.

```
class Laptop:
  def __init__(self,brand,price):
    self.brand= brand
    self.price=price
  def discount(self,price):
    self.price=price
    discount=10
    return  price-price*discount/100
b=Laptop("Apple",100000)
b.discount(100000)
```

```
90000.0
```

10.Create a Employee class with attributes for name and salary. Add a method to give a raise (increase the salary). Create an object and test the method.

```
class Employee:
  def __init__(self,name,salary):
    self.name=name
    self.salary=salary
  def raise_salary(self,salary):
    self.salary+=salary
    return self.salary
E1=Employee("Divya",45000)
E1.raise_salary(45000)
```

```
90000
```

11.Create a Movie class with attributes for title, director, and release year. Add a method to display the movie's information. Create an object and call the method.

```
class Movie:

  def __init__(self,title,director,release_year):
    self.title=title
    self.director=director
    self.release_year=release_year
  def information(self):
    info=(f"Title:{self.title}" " " f"Director:{self.director}" " " f"Release_year:{self.release_year}")
    return info
s1=Movie("abc","bnm",1998)
s1.information()
```

```
'Title:abc Director:bnm Release_year:1998'
```

12.Define a Product class with attributes for name and price. Add a method to calculate the price after tax (assume a fixed tax rate). Create an object and print the price after tax.

```
class Product:
  def __init__(self,name,price):
    self.name=name
    self.price=price
  def calculate(self,price):
    tax=2
    return price+price*tax/100

t1=Product("watch",5000)
t1.calculate(5000)
```

```
5100.0
```

13.Create a Player class with attributes for name and score. Add a method to update the score. Create an object and test the method

```
class Player:
  def __init__(self,name,score):
    self.name1=name
    self.score1=score

  def update(self,points):
    self.score1+=points
    return self.score1

P1=Player("Vinam",4)
P1.update(5)
```

```
9
```