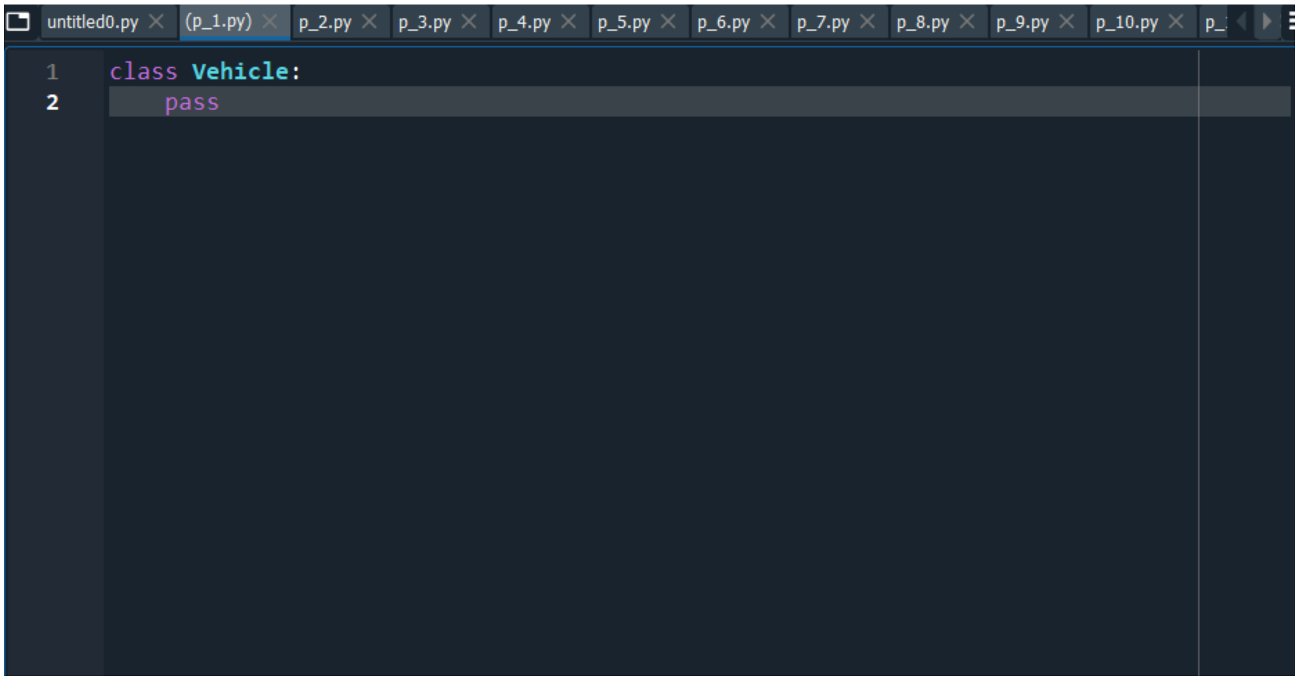
Assignment-2

Problems based on OOPS Concept in Python

1. Create a Vehicle class without any variables and methods

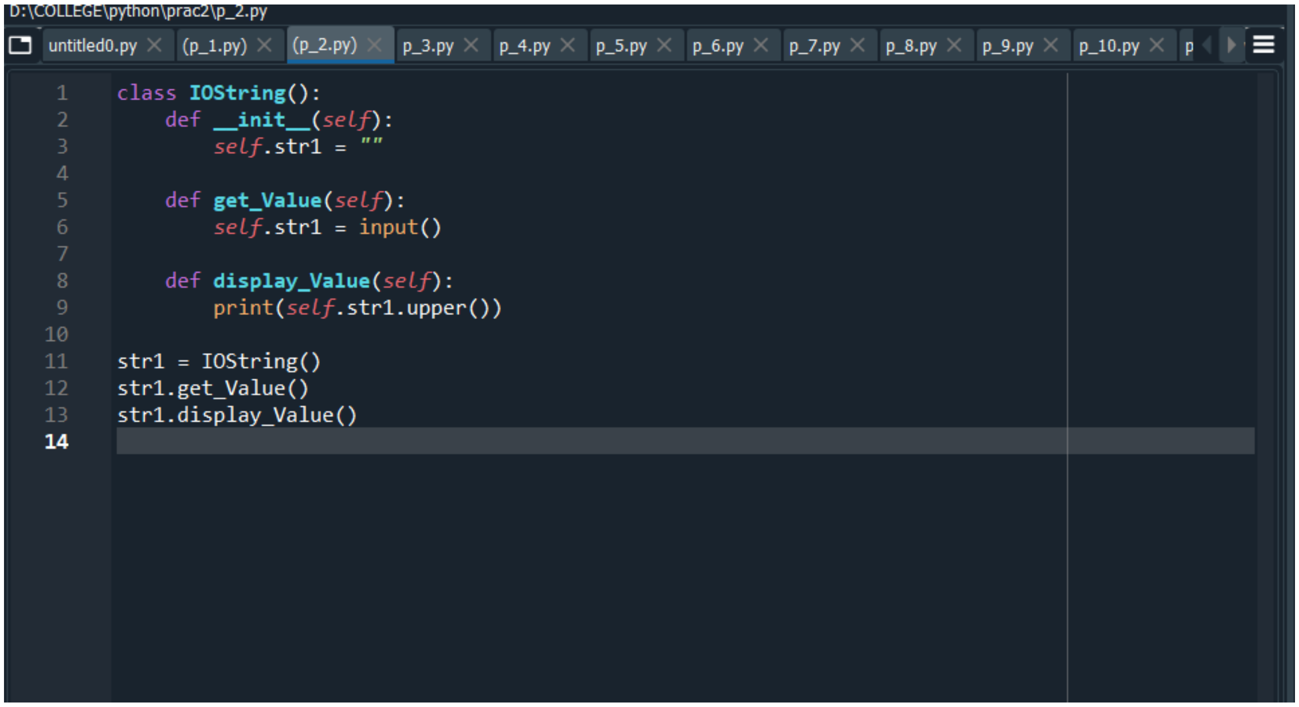
CODE:



```
class Vehicle:
    pass
```

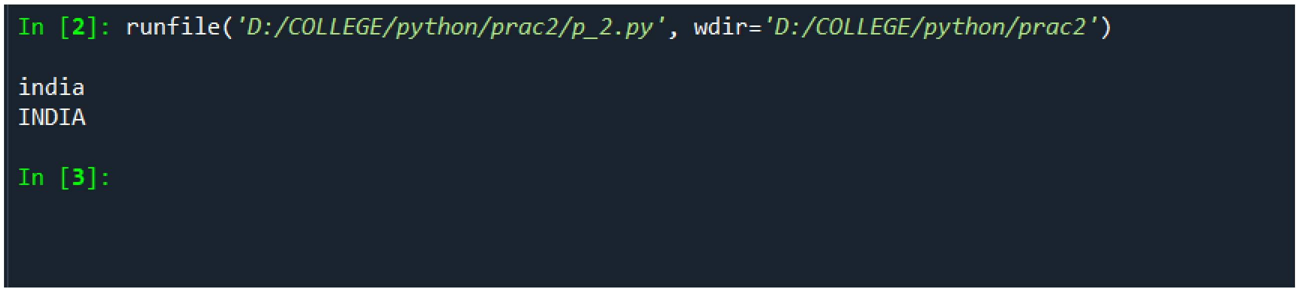
2. Write a Python class which has two methods get_Value and display_Value. get_Value accept a string from the user and display_Value print the string in upper case.

CODE:


D:\COLLEGE\python\prac2\p_2.py

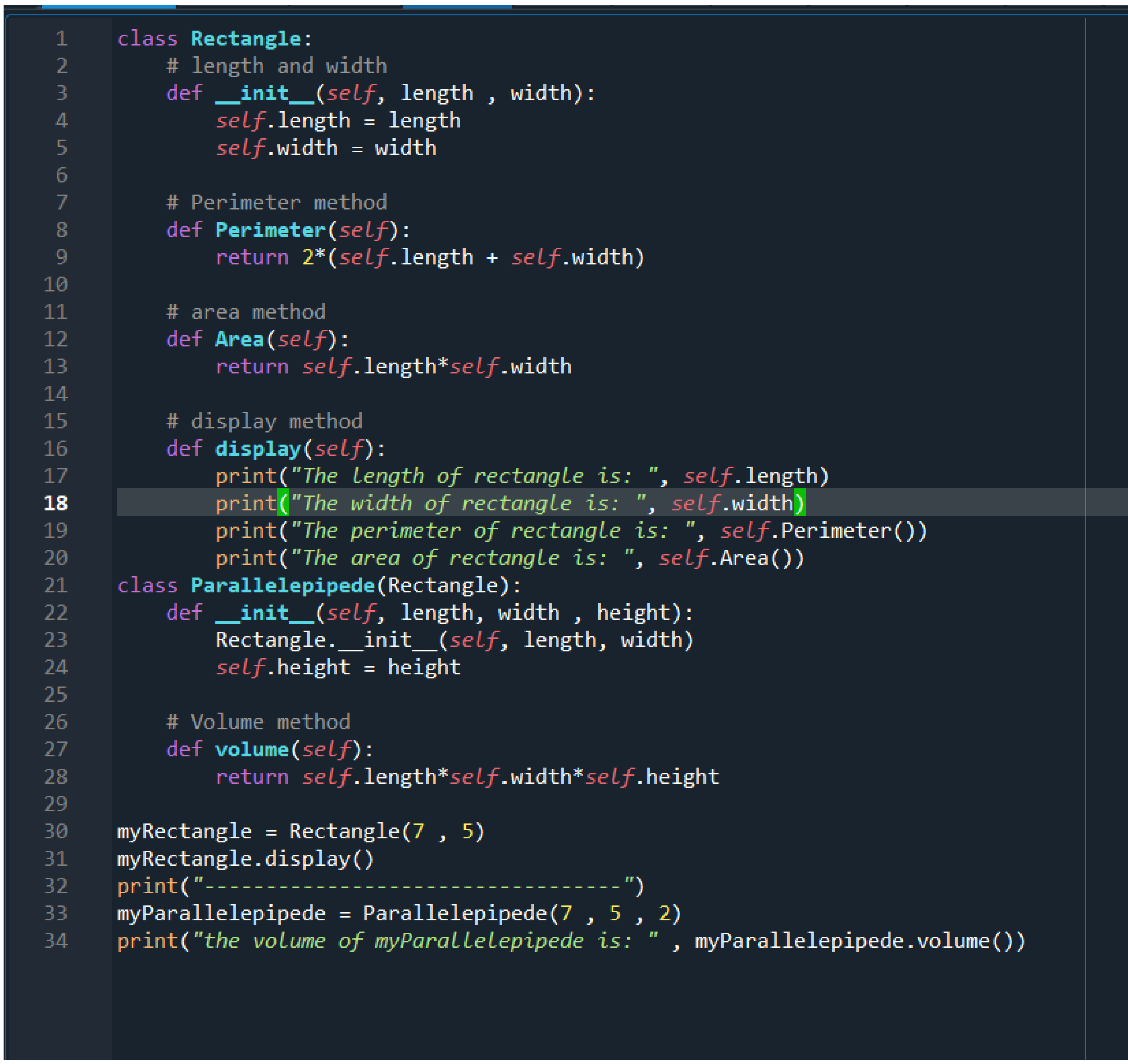
```python
class IOString():
    def __init__(self):
        self.str1 = ""

    def get_Value(self):
        self.str1 = input()

    def display_Value(self):
        print(self.str1.upper())

str1 = IOString()
str1.get_Value()
str1.display_Value()

```

OUTPUT:


```
In [2]: runfile('D:/COLLEGE/python/prac2/p_2.py', wdir='D:/COLLEGE/python/prac2')

india
INDIA

In [3]:
```

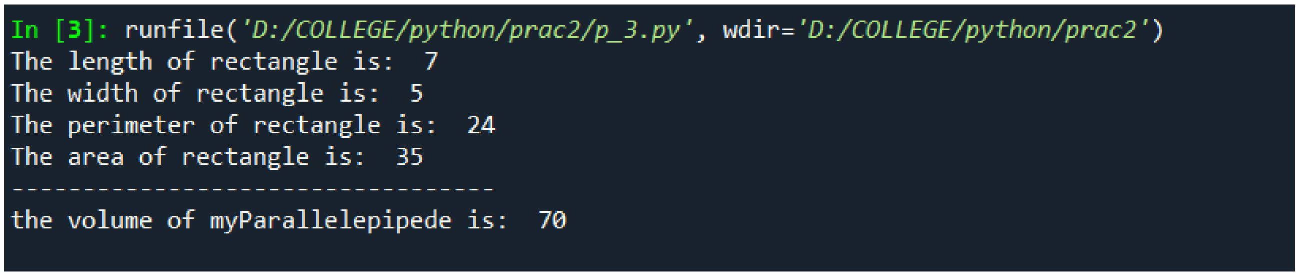
3. Write a **Rectangle class** in Python, allowing you to build a rectangle with **length** and **width** attributes. Create a **Perimeter()** method to calculate the perimeter of the rectangle and a **Area()** method to calculate the area of the rectangle. also Create a method **display()** that display the length, width, perimeter and area of an object created using an instantiation on rectangle class.
   - perimeter of Rectangle:2(l+w)
   - Area of Rectangle:l*w

CODE:



```python
class Rectangle:
    # length and width
    def __init__(self, length , width):
        self.length = length
        self.width = width

    # Perimeter method
    def Perimeter(self):
        return 2*(self.length + self.width)

    # area method
    def Area(self):
        return self.length*self.width

    # display method
    def display(self):
        print("The length of rectangle is: ", self.length)
        print("The width of rectangle is: ", self.width)
        print("The perimeter of rectangle is: ", self.Perimeter())
        print("The area of rectangle is: ", self.Area())
class Parallelepipede(Rectangle):
    def __init__(self, length, width , height):
        Rectangle.__init__(self, length, width)
        self.height = height

    # Volume method
    def volume(self):
        return self.length*self.width*self.height

myRectangle = Rectangle(7 , 5)
myRectangle.display()
print("----------------------------------")
myParallelepipede = Parallelepipede(7 , 5 , 2)
print("the volume of myParallelepipede is: " , myParallelepipede.volume())
```

OUTPUT:

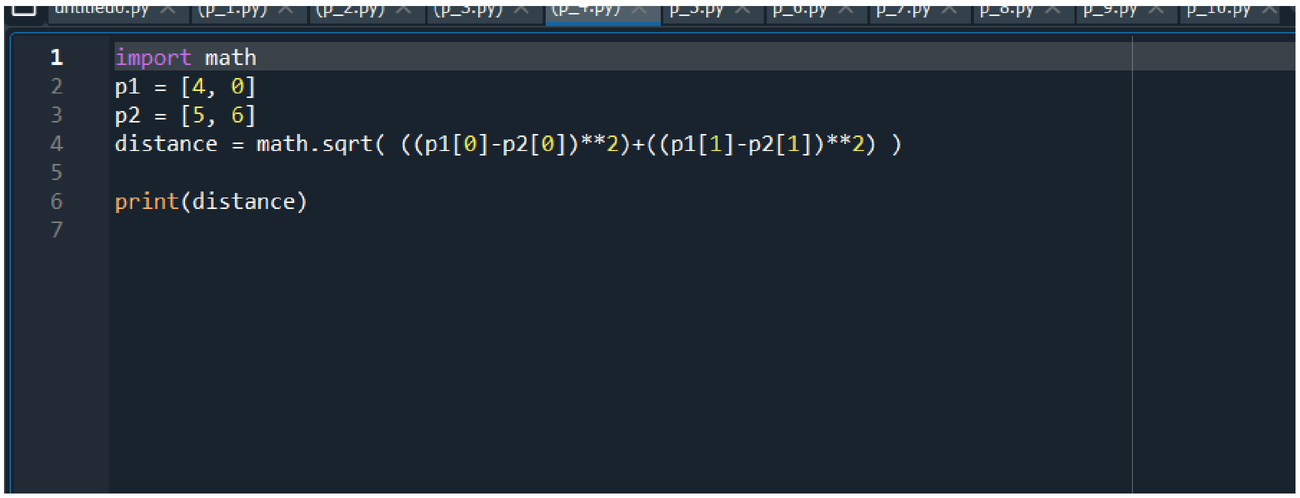

```
In [3]: runfile('D:/COLLEGE/python/prac2/p_3.py', wdir='D:/COLLEGE/python/prac2')
The length of rectangle is:  7
The width of rectangle is:  5
The perimeter of rectangle is:  24
The area of rectangle is:  35
----------------------------------
the volume of myParallelepipede is:  70
```

4. Create class as point. Objects from this class should have a method **show** to display the coordinates of the point, also have method **dist** that computes the distance between 2 points.

the distance between 2 points A(x1, y1) and B(x2, y2) can be compute

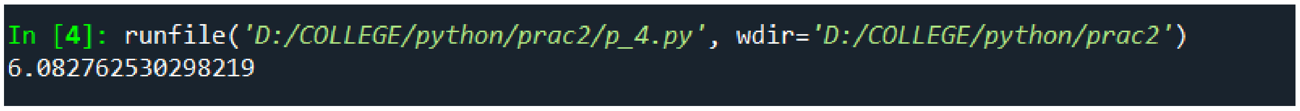
$$d(AB)= \sqrt{(x2-x1)^2+(y2-y1)^2}$$

CODE:



```python
import math
p1 = [4, 0]
p2 = [5, 6]
distance = math.sqrt( ((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2) )

print(distance)
```

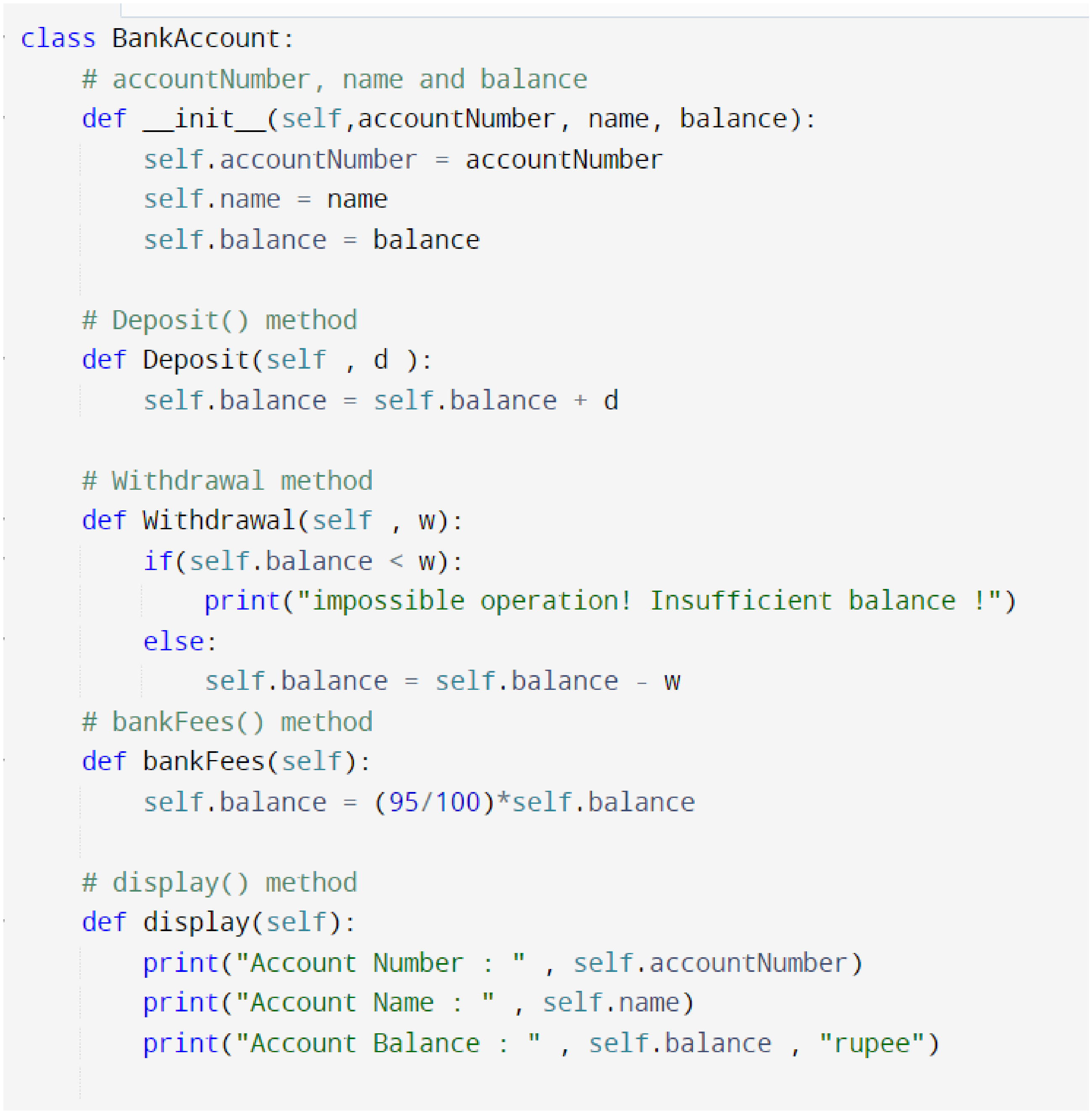
OUTPUT:



```
In [4]: runfile('D:/COLLEGE/python/prac2/p_4.py', wdir='D:/COLLEGE/python/prac2')
6.082762530298219
```

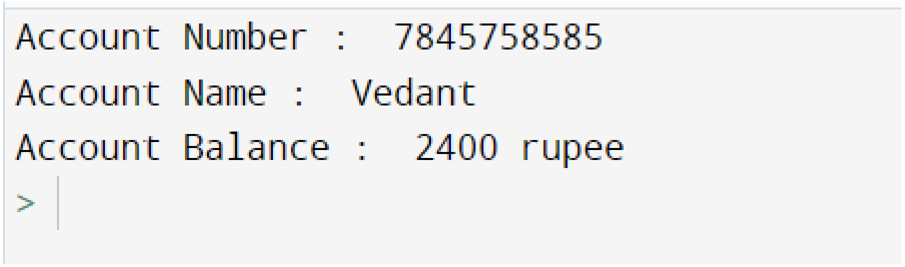
5. Create a Python class called **BankAccount** which represents a bank account, having a attributes: **accountNumber** (numeric type), **name** (name of the account owner as string type), balance. Create a **constructor** with parameters: **accountNumber, name, balance**.Create a **Deposit()** method which manages the deposit actions and display the updated balance. Create a **Withdrawal() method** which display the updated balance.

CODE:



```python
class BankAccount:
    # accountNumber, name and balance
    def __init__(self,accountNumber, name, balance):
        self.accountNumber = accountNumber
        self.name = name
        self.balance = balance

    # Deposit() method
    def Deposit(self , d ):
        self.balance = self.balance + d

    # Withdrawal method
    def Withdrawal(self , w):
        if(self.balance < w):
            print("impossible operation! Insufficient balance !")
        else:
            self.balance = self.balance - w
    # bankFees() method
    def bankFees(self):
        self.balance = (95/100)*self.balance

    # display() method
    def display(self):
        print("Account Number : " , self.accountNumber)
        print("Account Name : " , self.name)
        print("Account Balance : " , self.balance , "rupee")
```

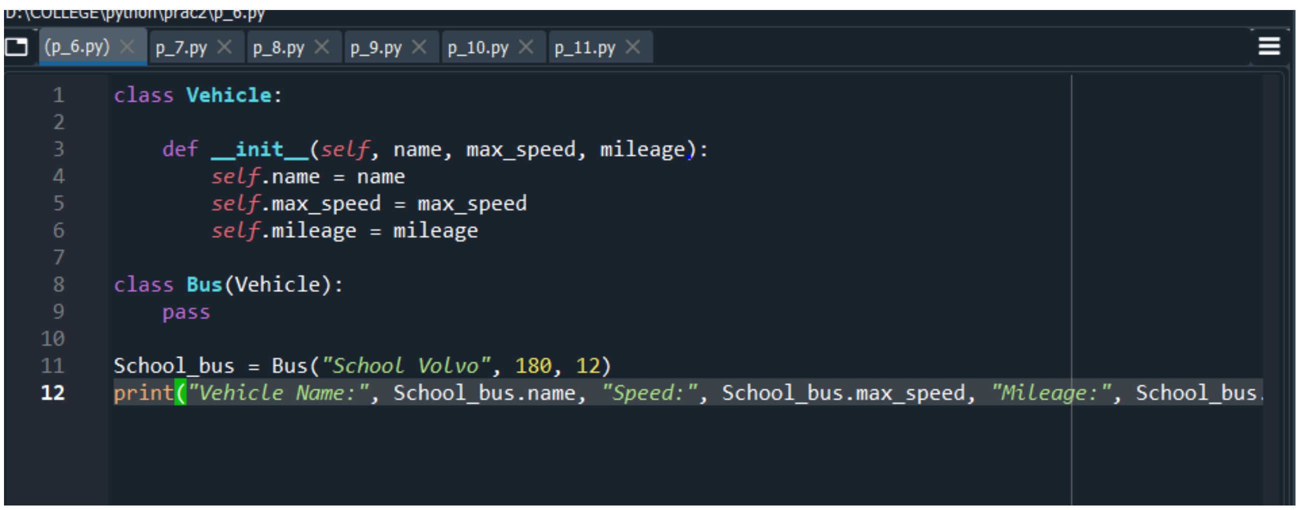
OUTPUT:



```
Account Number :  7845758585
Account Name :  Vedant
Account Balance :  2400 rupee
>
```

6. Create a child class Bus that will inherit al of the variables and methods of the Vehicle class.

CODE:

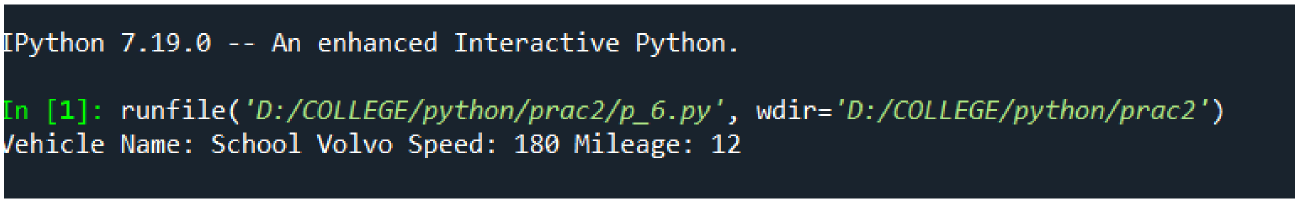


D:\COLLEGE\python\prac2\p_6.py

(p_6.py) p_7.py p_8.py p_9.py p_10.py p_11.py

```python
class Vehicle:

    def __init__(self, name, max_speed, mileage):
        self.name = name
        self.max_speed = max_speed
        self.mileage = mileage

class Bus(Vehicle):
    pass

School_bus = Bus("School Volvo", 180, 12)
print("Vehicle Name:", School_bus.name, "Speed:", School_bus.max_speed, "Mileage:", School_bus.
```

OUTPUT:



```
IPython 7.19.0 -- An enhanced Interactive Python.

In [1]: runfile('D:/COLLEGE/python/prac2/p_6.py', wdir='D:/COLLEGE/python/prac2')
Vehicle Name: School Volvo Speed: 180 Mileage: 12
```

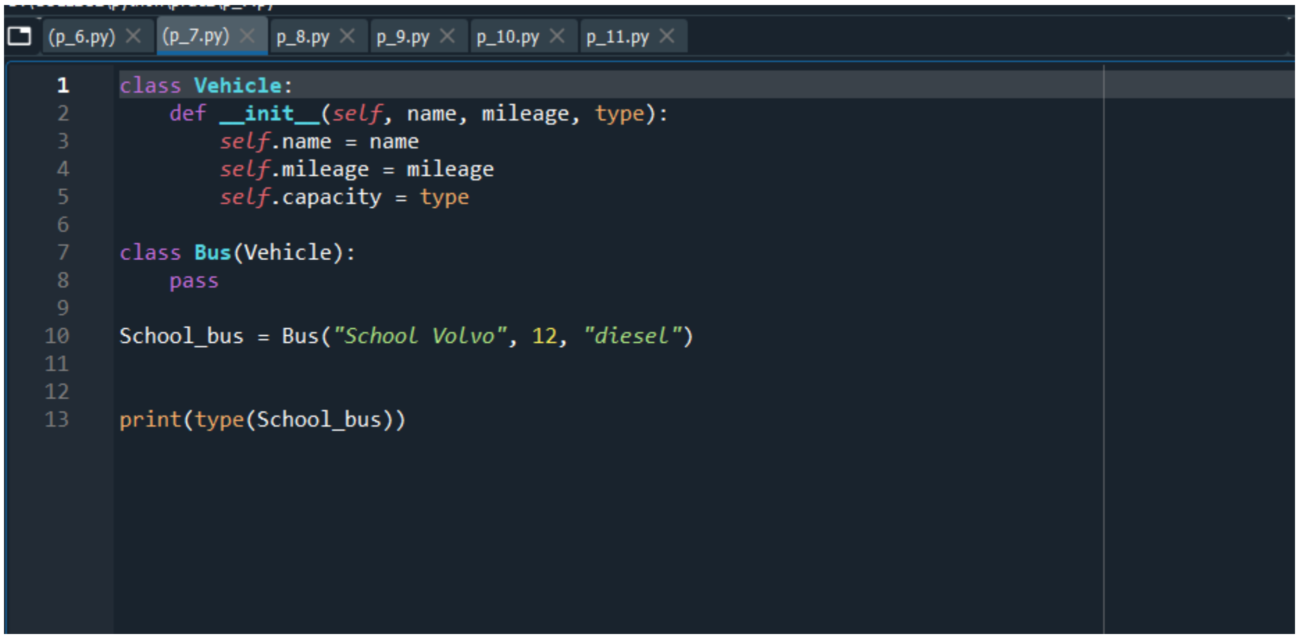
7. **Determine which class a given Bus object belongs to (Check type of an object).**

Given:

```
class Vehicle:

    def_init_(self, name, mileage, type):

        self.name = name

        self.mileage = mileage

        self.type = type


class Bus(Vehicle):

    pass
```

b = Bus("School Volvo", 12, “diesel”)

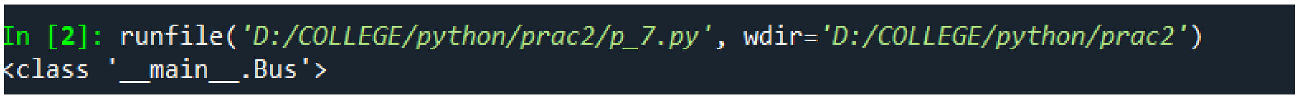
CODE:



```
(p_6.py)  (p_7.py)  p_8.py  p_9.py  p_10.py  p_11.py
```

```python
class Vehicle:
    def __init__(self, name, mileage, type):
        self.name = name
        self.mileage = mileage
        self.capacity = type

class Bus(Vehicle):
    pass

School_bus = Bus("School Volvo", 12, "diesel")


print(type(School_bus))
```

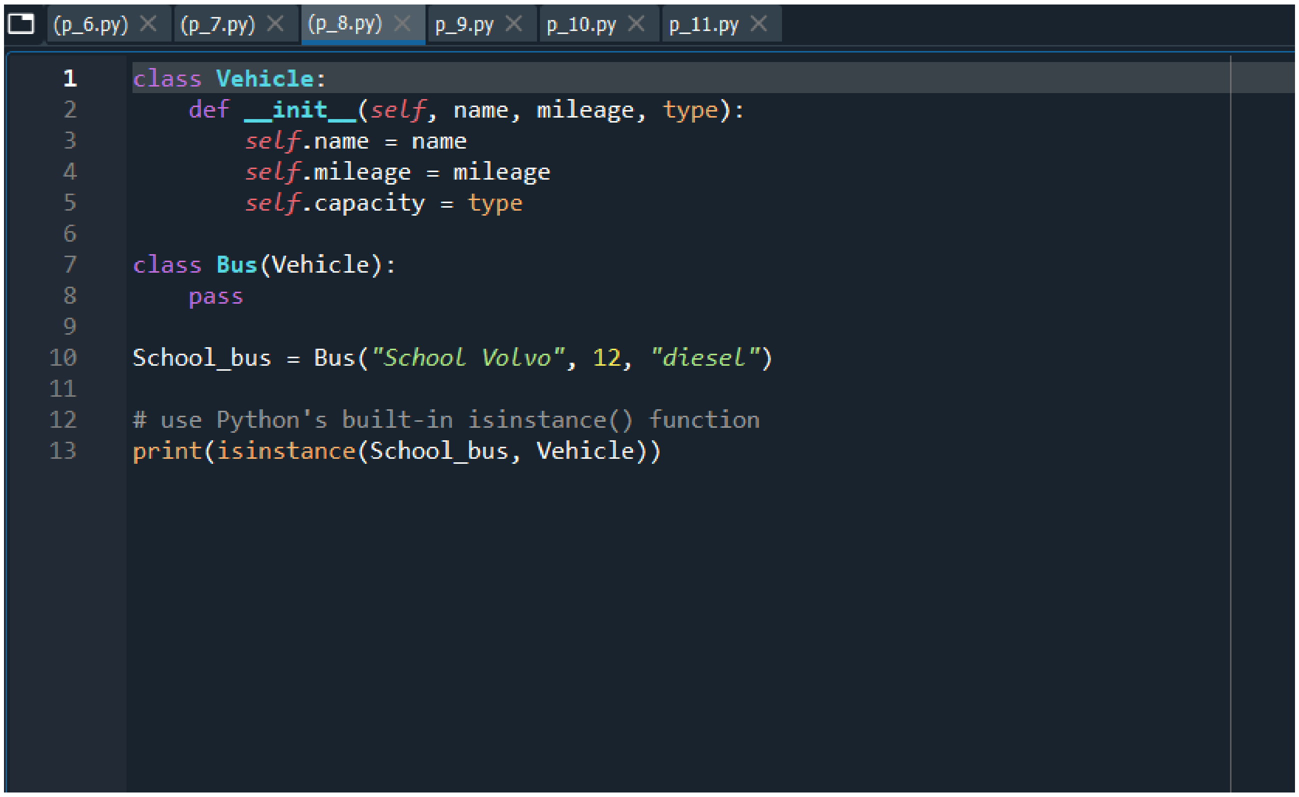
OUTPUT:



```
In [2]: runfile('D:/COLLEGE/python/prac2/p_7.py', wdir='D:/COLLEGE/python/prac2')
<class '__main__.Bus'>
```

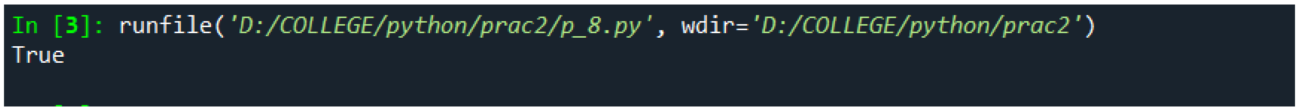
8. **In above Given Code, Determine if object of bus class is an instance of the Vehicle class.**

   CODE:



```python
class Vehicle:
    def __init__(self, name, mileage, type):
        self.name = name
        self.mileage = mileage
        self.capacity = type

class Bus(Vehicle):
    pass

School_bus = Bus("School Volvo", 12, "diesel")

# use Python's built-in isinstance() function
print(isinstance(School_bus, Vehicle))
```

OUTPUT:



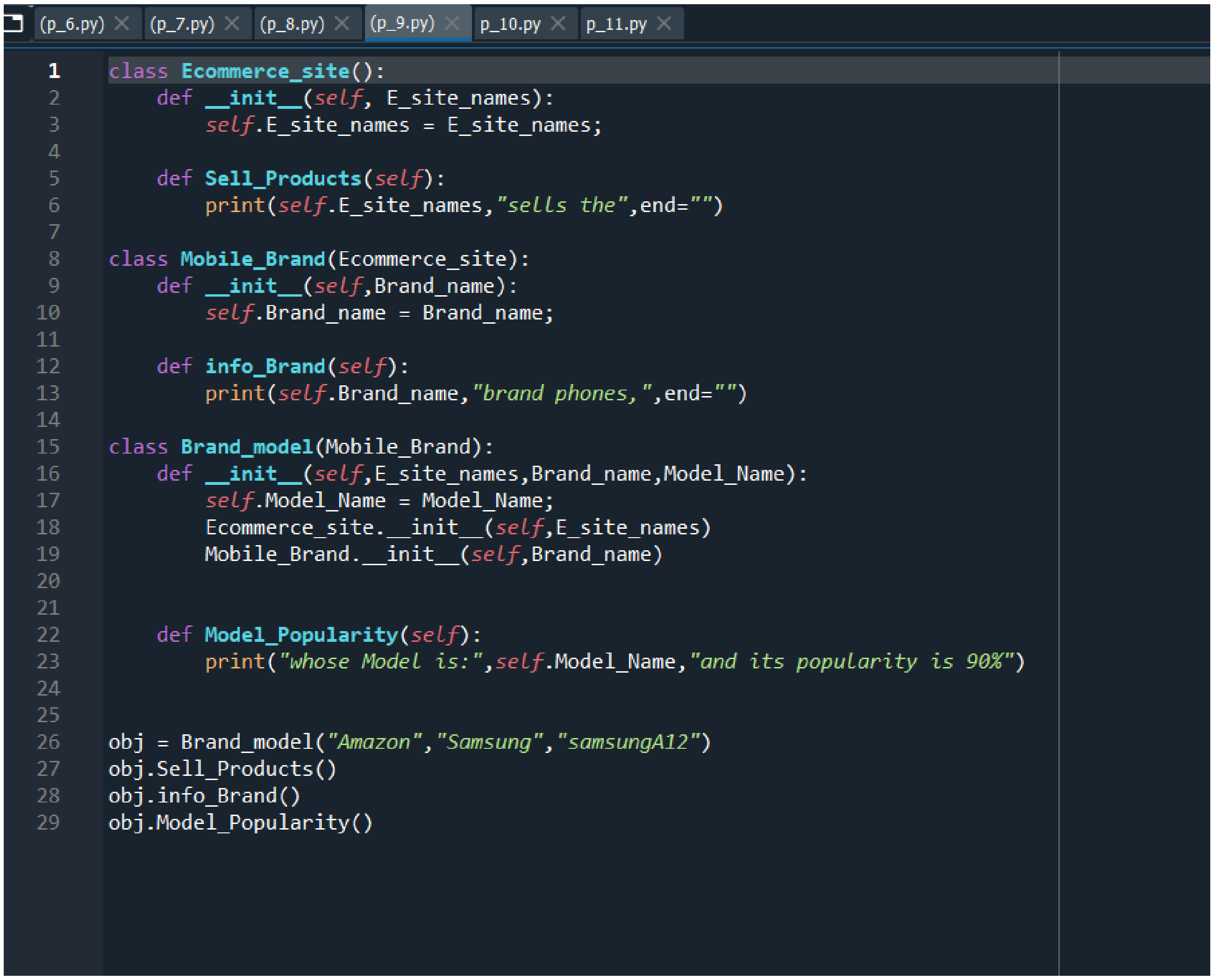
```
In [3]: runfile('D:/COLLEGE/python/prac2/p_8.py', wdir='D:/COLLEGE/python/prac2')
True
```

9. **Create the class Ecommerce-site. Define its attributes E_site_names and the method as Sell_Products(). This class is inherited by Class Mobile_Brand. Which will inherit all the properties and method of the Ecommerce-site. This Derived class has additional Attribute and the Method like Brand_name and info_Brand(). create grand Child class Brand_model which you have to derive from class Mobile_Brand. That has Attribute as Model_Name and Method as Model_Popularity().**

Display the output as:

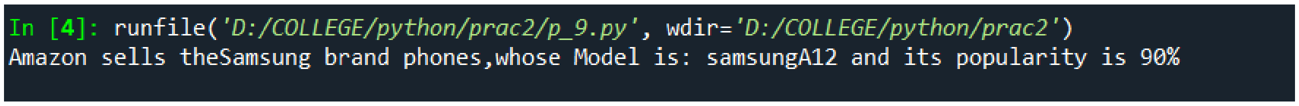
Ecommerce-Site: Amazon Sell the Samsung Brand phones, whose Model is: samsungA12 and its popularity is 90%

CODE:



```
class Ecommerce_site():
    def __init__(self, E_site_names):
        self.E_site_names = E_site_names;

    def Sell_Products(self):
        print(self.E_site_names,"sells the",end="")

class Mobile_Brand(Ecommerce_site):
    def __init__(self,Brand_name):
        self.Brand_name = Brand_name;

    def info_Brand(self):
        print(self.Brand_name,"brand phones,",end="")

class Brand_model(Mobile_Brand):
    def __init__(self,E_site_names,Brand_name,Model_Name):
        self.Model_Name = Model_Name;
        Ecommerce_site.__init__(self,E_site_names)
        Mobile_Brand.__init__(self,Brand_name)


    def Model_Popularity(self):
        print("whose Model is:",self.Model_Name,"and its popularity is 90%")


obj = Brand_model("Amazon","Samsung","samsungA12")
obj.Sell_Products()
obj.info_Brand()
obj.Model_Popularity()
```

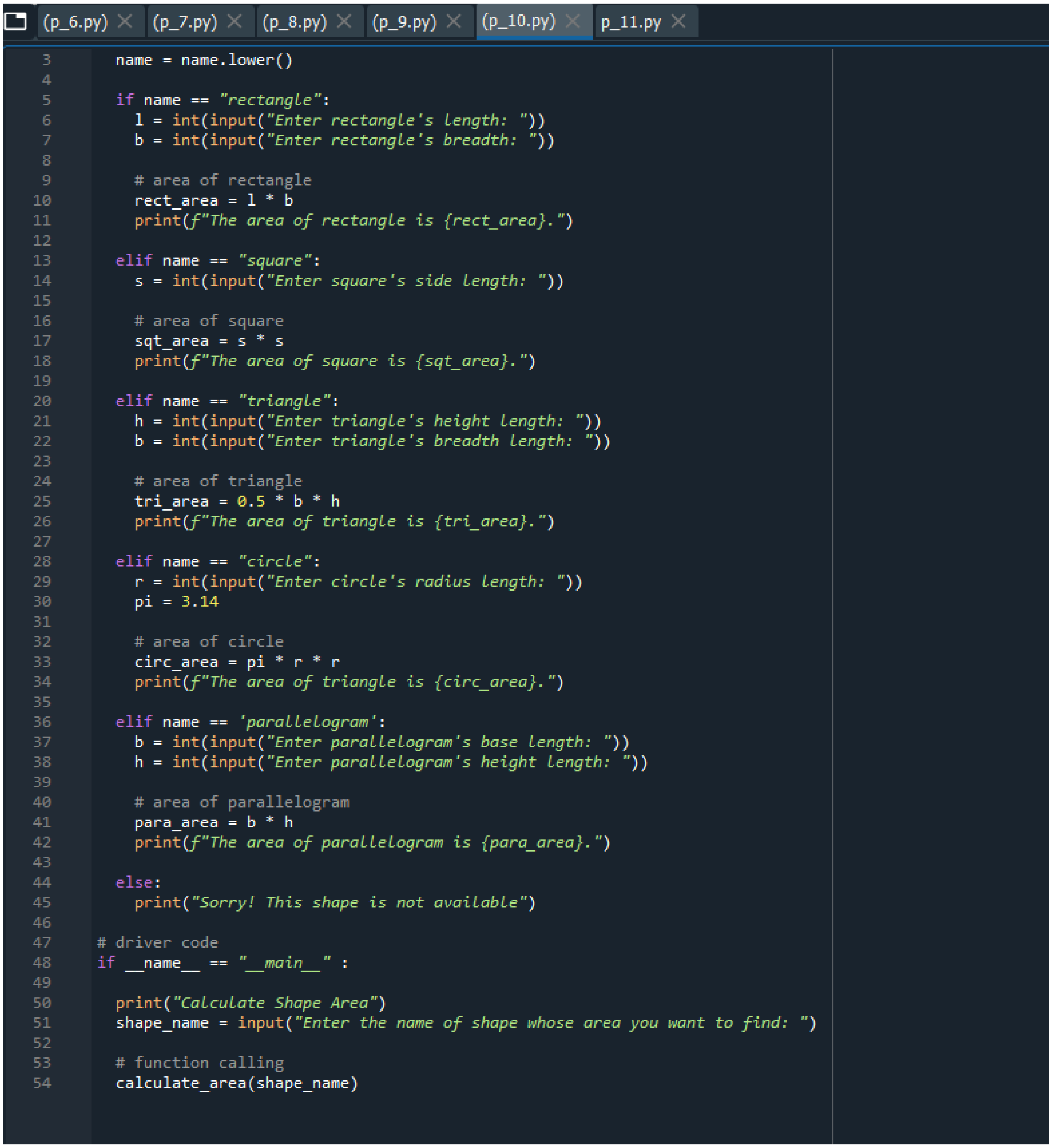
OUTPUT:



```
In [4]: runfile('D:/COLLEGE/python/prac2/p_9.py', wdir='D:/COLLEGE/python/prac2')
Amazon sells theSamsung brand phones,whose Model is: samsungA12 and its popularity is 90%
```

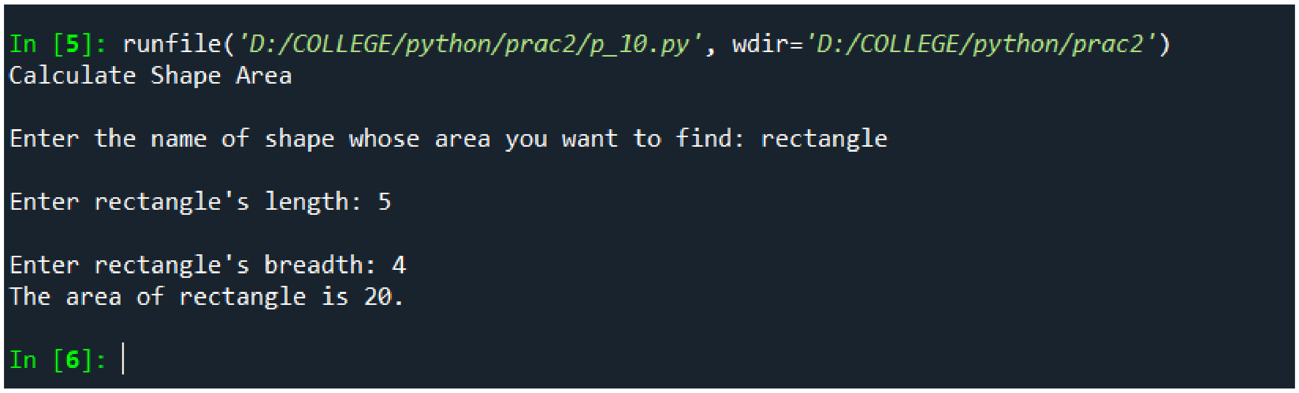
10. Demonstrate the concept of overriding by calculating area of Shape (Circle, Square, triangle).

CODE:



```
  name = name.lower()

  if name == "rectangle":
    l = int(input("Enter rectangle's length: "))
    b = int(input("Enter rectangle's breadth: "))

    # area of rectangle
    rect_area = l * b
    print(f"The area of rectangle is {rect_area}.")

  elif name == "square":
    s = int(input("Enter square's side length: "))

    # area of square
    sqt_area = s * s
    print(f"The area of square is {sqt_area}.")

  elif name == "triangle":
    h = int(input("Enter triangle's height length: "))
    b = int(input("Enter triangle's breadth length: "))

    # area of triangle
    tri_area = 0.5 * b * h
    print(f"The area of triangle is {tri_area}.")

  elif name == "circle":
    r = int(input("Enter circle's radius length: "))
    pi = 3.14

    # area of circle
    circ_area = pi * r * r
    print(f"The area of triangle is {circ_area}.")

  elif name == 'parallelogram':
    b = int(input("Enter parallelogram's base length: "))
    h = int(input("Enter parallelogram's height length: "))

    # area of parallelogram
    para_area = b * h
    print(f"The area of parallelogram is {para_area}.")

  else:
    print("Sorry! This shape is not available")

# driver code
if __name__ == "__main__" :

  print("Calculate Shape Area")
  shape_name = input("Enter the name of shape whose area you want to find: ")

  # function calling
  calculate_area(shape_name)
```

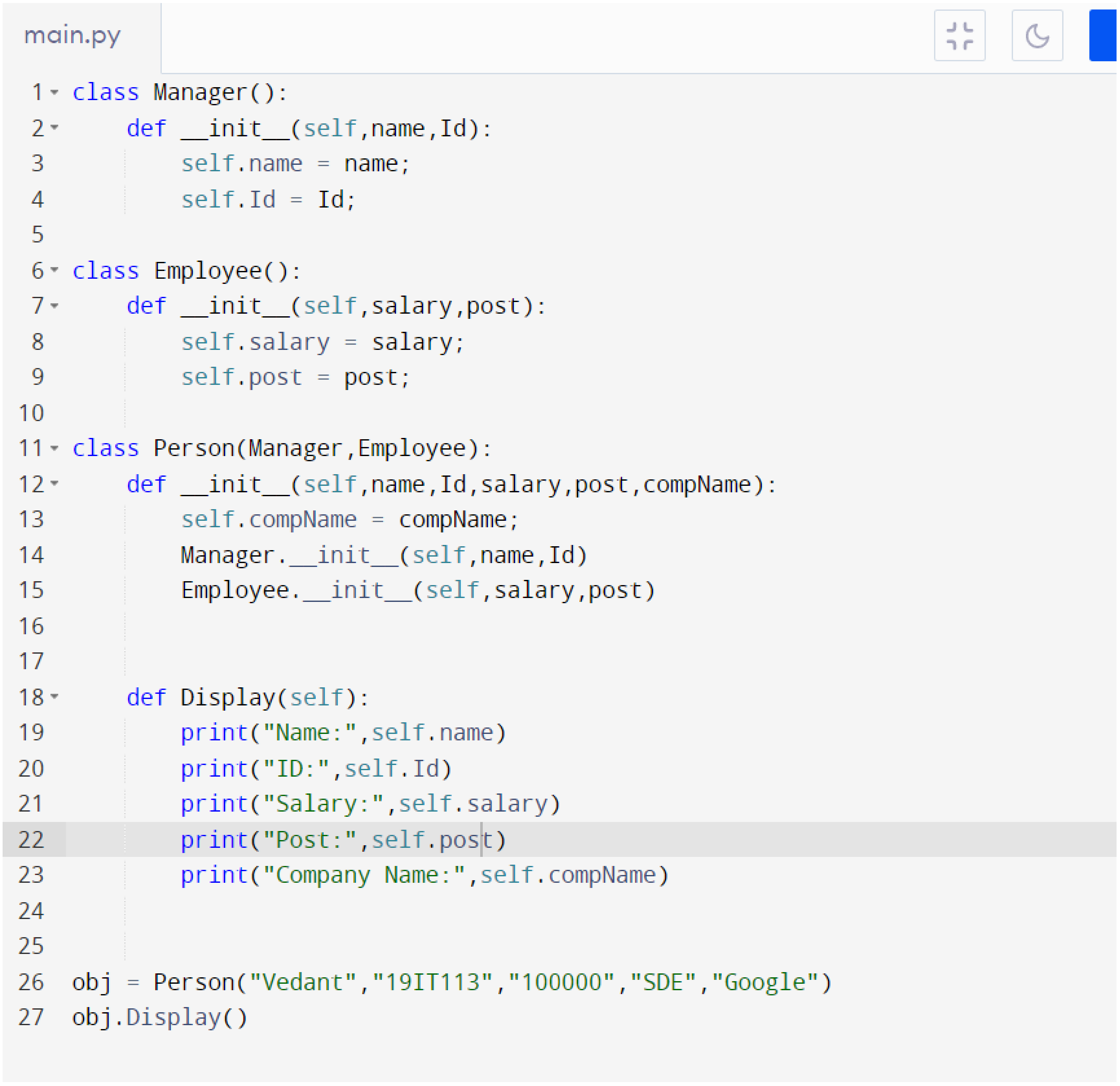
OUTPUT:



```
In [5]: runfile('D:/COLLEGE/python/prac2/p_10.py', wdir='D:/COLLEGE/python/prac2')
Calculate Shape Area

Enter the name of shape whose area you want to find: rectangle

Enter rectangle's length: 5

Enter rectangle's breadth: 4
The area of rectangle is 20.

In [6]: 
```

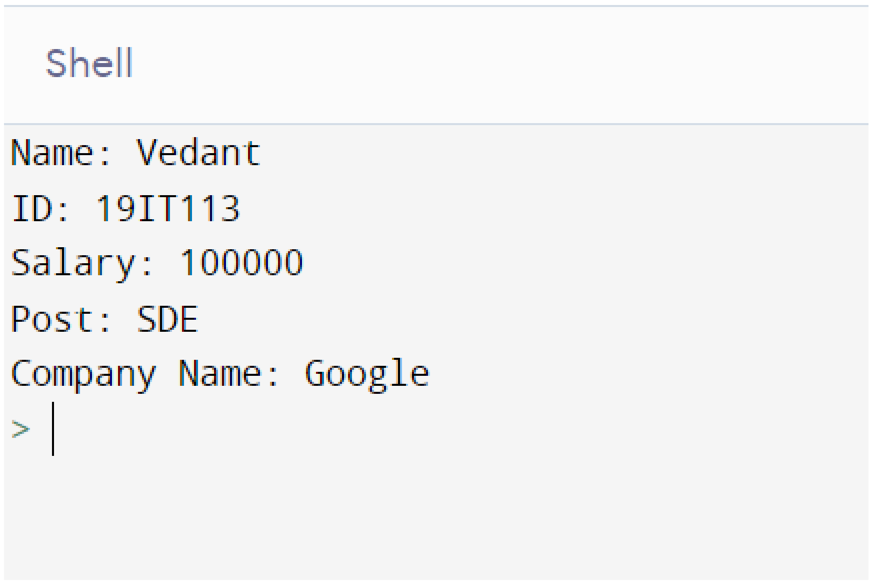
11. Create the Class Person which will inherits from class Manager and Class employee. Class Manager have the Attributes Name and id. Class employee have attributes salary and post. Class Person which is inherited from both these classes have an additional Parameter: Company Name and the Display Method: which display all the information Regarding that Person

CODE:



```python
class Manager():
    def __init__(self,name,Id):
        self.name = name;
        self.Id = Id;

class Employee():
    def __init__(self,salary,post):
        self.salary = salary;
        self.post = post;

class Person(Manager,Employee):
    def __init__(self,name,Id,salary,post,compName):
        self.compName = compName;
        Manager.__init__(self,name,Id)
        Employee.__init__(self,salary,post)


    def Display(self):
        print("Name:",self.name)
        print("ID:",self.Id)
        print("Salary:",self.salary)
        print("Post:",self.post)
        print("Company Name:",self.compName)


obj = Person("Vedant","19IT113","100000","SDE","Google")
obj.Display()
```

OUTPUT:


Shell

```
Name: Vedant
ID: 19IT113
Salary: 100000
Post: SDE
Company Name: Google
> 
```