ROHIT GUPTA

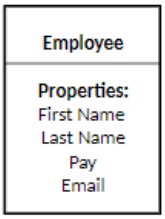
500087112

BATCH=27

Experiment 10&11

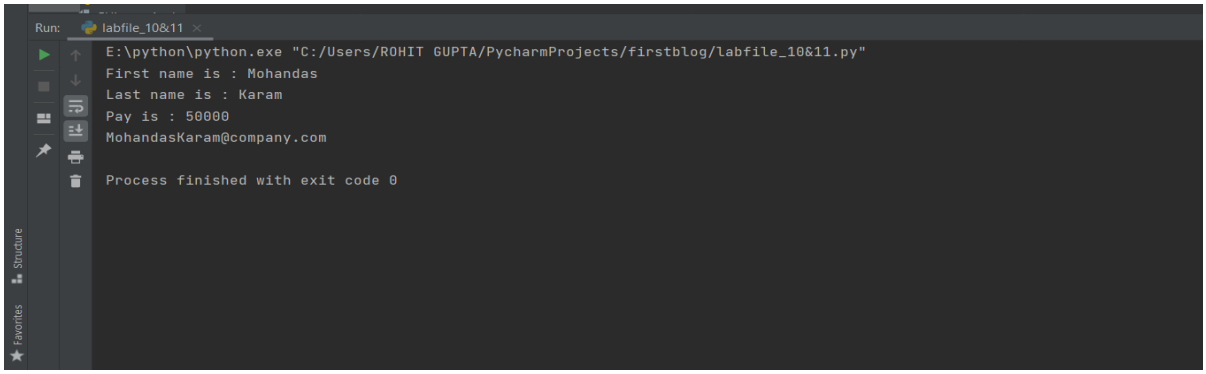
Question 1:

- Create a class Employee with following properties
  - First Name
  - Last Name
  - Pay
  - Email : should be automatically generated as
    - Firstname + '.' + Lastname + "@company.com"
- Test the code with following information of an Employee:
  - First name is : ***Mohandas***
  - Last name is : ***Gandhi***
  - *Pay is : **50000***



Answer:

```
class Employee:
    def __init__(self,firstname,lastname,pay):
        self.firstname=firstname
        self.lastname=lastname
        self.pay=pay
    def email(self):
        return f"{self.firstname}{self.lastname}@company.com"
    def details(self):
        print(f"First name is : {self.firstname}")
        print(f"Last name is : {self.lastname}")
        print(f"Pay is : {self.pay}")
name1=Employee("Mohandas","Karam",50000)
name1.details()
print(name1.email())
```



QUESTION 2: Q2. Perform the following instructions:

a) Create a Vehicle class with max_speed and mileage as instance attributes. Additionally, create a method named seating_capacity() using the below syntax:

```
def seating_capacity(self, capacity):
    return f"The seating capacity of a {self.name} is {capacity} passengers"
```

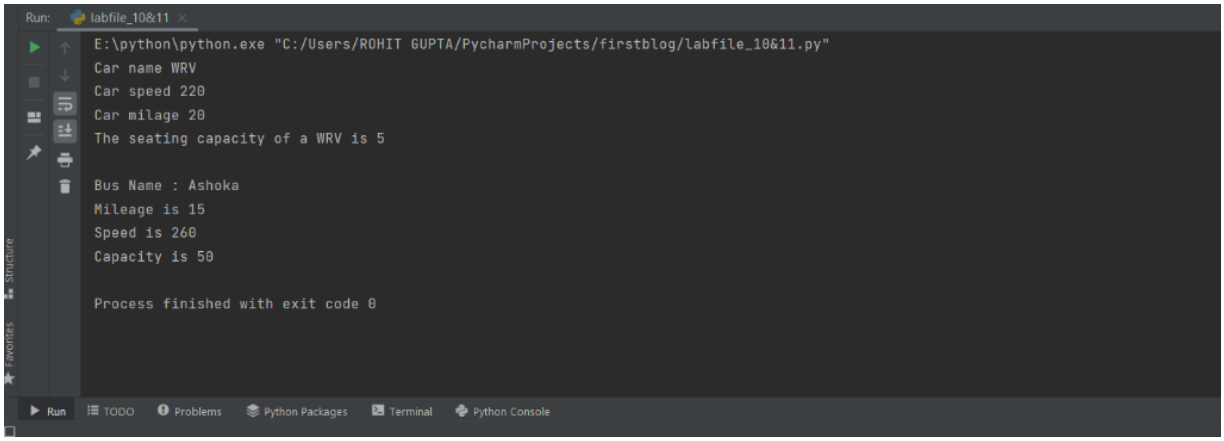
b) Create child class 'Bus' that will inherit all of the variables and methods of the Vehicle class. Set the seating capacity of the bus to 50 using super().

c) Create a Bus object that will inherit all of the variables and methods of the Vehicle class and display it.

d) Define a class attribute "color" with a default value white. I.e., Every Vehicle should be white.

Answer:

```
class vehicle:
    def __init__(self,name,speed,milage):
        self.speed=speed
        self.name=name
        self.milage=milage

    def seating_capacity(self,capacity):
        self.capacity=capacity
        return f"The seating capacity of a {self.name} is
{capacity}"

class Bus(vehicle):
    def __init__(self,name,speed,milage):
        super().__init__(name,speed,milage)
        super().seating_capacity(50)

    def printdetails(self):
        print(f"\nBus Name : {self.name}")
        print(f"Mileage is {self.milage}")
        print(f"Speed is {self.speed}")
        print(f"Capacity is {self.capacity}")

car=vehicle("WRV",220,20)
print("Car name",car.name)
print("Car speed",car.speed)
print("Car milage",car.milage)
print(car.seating_capacity(5))

bus1=Bus("Ashoka",260,15)
bus1.printdetails()
```



QUESTION 3:

```
class Account:
    def __init__(self, initial_amount):
        self.balance = initial_amount
    def withdraw(self,amount):
        self.balance = self.balance - amount
    def deposit(self,amount):
        self.balance = self.balance + amount
ac = Account(1000)
ac.balance = 2000 #stmt1
ac.balance = -1000 #stmt2
print(ac.balance) #stmt3
```

Answer:

```
class Account:
    def __init__(self,Current_amount):
        self.balance=Current_amount
        print(f"Accounted created with amount {self.balance}")
    def withdraw(self,amount):
        self.amount=amount
        self.balance=self.balance-self.amount
        print(f"\nCurrent balance is {self.balance}")
        print(f"Withdraw amount is {self.amount}")
    def deposit(self,amount):
        self.amount=amount
        self.balance=self.balance+self.amount
        print(f"\nCurrent balance is {self.balance}")
        print(f"Deposited amount is {self.amount}")

ac=Account(10000)
```

```
ac.withdraw(2000)
ac.deposit(5000)
```