UPES PYTHON LAB ASSIGNMENT

Experiment 10&11

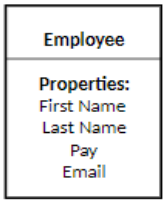
NAME: Anmol yadav

SAP ID: 500083814

BATCH: B8

QUESTION NO: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***



SOL :

```python
class Employee():
    def __init__(self,FirstName,LastName,Pay):
        self.FirstName=FirstName
        self.LastName=LastName
        self.Pay=Pay

    def Email(self):
        return self.FirstName +"."+ self.LastName+"@company.com"

Empl=Employee("Mohandas", "Gandhi", 50000)
print("Fist Name :", Empl.FirstName)
print("List Name :", Empl.LastName)
print("Pay :",Empl.Pay)

print("email :", Empl.Email())
```

OUTPUT:-

```
Fist Name : Mohandas
List Name : Gandhi
Pay : 50000
email : Mohandas.Gandhi@company.com
```

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:

```
defseating_capacity(self, capacity):
returnf"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.

SOL:

```python
class Vehicle():
    color = "white"

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

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


class Bus(Vehicle):
    def seating_capacity(self, capacity):
        return super().seating_capacity(50)

    def display(self):
        print(self.name)


b1 = Bus('TATA bus', 120, 3.63)
print(b1.seating_capacity(50))
b1.display()
print(Vehicle.color)
```

OUTPUT:

```
The seating capacity of a TATA bus is 50 passengers
TATA bus
white
```

QUESTION 3:

Q3.

Q: List the risk associated with the implementation of Account class. Suggest a solution.

```
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
```

SOL:

```
class account:
    def __init__(self, intial_amount):
        self.balance=intial_amount
    def withdraw(self,amount):
        if amount>0:
            if self.balance>=amount:
                self.balance=self.balance-amount
            else:
                print("Insufficent balance")
        else:
            print("Invalid withdraw input ")
    def deposit(self,amount):
        if amount>0:
            self.balance=self.balance+amount
        else:
            print("Invalid deposit input")

ac=account(1000)
ac.withdraw(2000)
ac.deposit(-1000)
print(ac.balance)
```

OUTPUT:

```
Insufficent balance
Invalid deposit input
1000
```