Assignment

PW SKILLS

Q1, Create a vehicle class with an init method having instance variables as name_of_vehicle, max_speed and average_of_vehicle.

Ans.

```
class vehicle():

    def __init__(self,name_of_vehicle,max_speed,average_of_vehicle):
        self.name_of_vehicle=name_of_vehicle
        self.max_speed=max_speed
        self.average_of_vehicle=average_of_vehicle
```

Q2. Create a child class car from the vehicle class created in Que 1, which will inherit the vehicle class. Create a method named seating_capacity which takes capacity as an argument and returns the name of the vehicle and its seating capacity.

Ans.

```
class vehicle():

    def __init__(self,name_of_vehicle,max_speed,average_of_vehicle):
        self.name_of_vehicle=name_of_vehicle
        self.max_speed=max_speed
        self.average_of_vehicle=average_of_vehicle

class car(vehicle):

    def seating_capacity(self,capacity):
        self.capacity=capacity
        print(f"The seating capacity is: {capacity}")
        print(f"The name of the vehicle is:{self.name_of_vehicle}")

obj_car=car("Inova",400,15)
obj_car.seating_capacity(7)
```

OUTPUT:

```
The seating capacity is: 3
The name of the vehicle is:Inova
```

Q3. What is multiple inheritance? Write a python code to demonstrate multiple inheritance.

Ans: *When a class is derived from more than one Parent class it is called multiple Inheritance.*

Example:

```
class class1:
    def test_class1(self):
        print("This is Class1")

class class2(class1):
    def test_class2(self):
        print("This is Class2")

class class3(class2):
    def test_class3(self):
        print("This is Class3")

obj_class3=class3()
obj_class3.test_class1()
```

OUTPUT:

```
This is Class1
```

Q4. What are getter and setter in python? Create a class and create a getter and a setter method in this class.

Ans: *The methods used in Object-Oriented Programming (OOPS) which helps to access the private attributes from a class are called getters. Setters are the methods used in OOPS feature which helps to set the value to private attributes in a class.*

```
class car1:
    def __init__(self, year, make,model,speed):
        self.__year= year
        self.__make=make
        self.__model=model
        self.__speed=speed

    def set_speed(self,speed):
        self.__speed= 0 if speed<0 else speed  #setters to set the values of private attribute speed

    def get_speed(self):
        return self.__speed #Getters to get the values of private attribute speed
```

Q5.What is method overriding in python? Write a python code to demonstrate method overriding.

Ans: *Method overriding describes the Polymorphism feature of object-oriented programming language where the subclass or child class can provide the program with specific characteristics or a specific implementation process of data provided that are already defined in the parent class.*

Example:

```
class data_science():
    def syllabus(self):
        print("This is data science Syllabus")


class cpp():
    def syllabus(self):
        print("This is C++ Syllabus")


def class_overriding(class_object):
    for i in class_object:
        i.syllabus()
obj_data_science = data_science()
obj_cpp = cpp()
class_obj = [obj_data_science, obj_cpp]
class_overriding (class_obj)
```

OUTPUT:

```
This is data science Syllabus
This is C++ Syllabus
```