```
                          Name-Rakesh Rajput
```

In [2]:
```
"""
Que 1 Create a vehicle class with an init method having instance variables as
name_of_vehicle, max_speed and average_of_vehicle.
"""

class vehical:
    def __init__(self,name_of_vehical,max_speed,average_of_vehical):
        self.name_of_vehical=name_of_vehical
        self.max_speed=max_speed
        self.average_of_vehical=average_of_vehical

    def vehical_detail(self):
        return self.average_of_vehical,self.max_speed,self.average_of_vehical
```

In [5]:
```
"""
Que 1-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.
"""

class vehicle:
    def seating_capacity(self):
        return name_vehicle,seating_capacity
class car(vehicle):
    pass
```

In [7]:
```
"""
Que-1. What is multiple inheritance? Write a python code to demonstrate
multiple inheritance.

Ans 1-
Multiple inheritance-Multiple inheritance is a feature of some object-oriented
computer programming languages in which an object or class can inherit features
from more than one parent object or parent class
"""
class class1:
    def test_class1(self):
        return 'aur dude'


class class2:
    def test_class2(self):
        return 'hi guys'


class class3(class1,class2):
    pass

obj1=class3()
```

In [8]:
```
obj1.test_class1()
```

Out[8]:
```
'hi guys'
```

In [9]:
```
obj1.test_class2()
```

Out[9]:
```
'hi guys'
```

In [10]:
```
"""
Que 4-
What are getter and setter in python? Create a class and create a getter
and a setter method in this class.

Ans 4-Getters: These are the methods used in Object-Oriented Programming (OOPS)
which helps to access the private attributes from a class. Setters: These are
the methods used in OOPS feature which helps to set the value to private
attributes in a class

"""


class pythonpoint:
    def __init__(self, age = 0):
         self._age = age
      # using the getter method
    def get_age(self):
        return self._age
      # using the setter method
    def set_age(self, a):
        self._age = a

John = pythonpoint()

#using the setter function
John.set_age(19)

# using the getter function
print(John.get_age())

print(John._age)
```

```
19
19
```

In [11]:
```
"""
Que 5-What is method overriding in python? Write a python code to
demonstrate method overriding.

Ans 5-Method overriding is a feature of object-oriented programming languages
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 or superclass
"""


class Parent():

    # Constructor
    def __init__(self):
        self.value = "Inside Parent"

    # Parent's show method
    def show(self):
        print(self.value)

# Defining child class
class Child(Parent):

    # Constructor
    def __init__(self):
        self.value = "Inside Child"

    # Child's show method
    def show(self):
        print(self.value)


# Driver's code
obj1 = Parent()
obj2 = Child()

obj1.show()
obj2.show()
```

```
Inside Parent
Inside Child
```

In [ ]: