In [3]:

```python
# Declaration of class
# Instance, Static, Local
class Vehicle:
#    def __init__(self):
#       self.color="White"
#       self.make="Maruti"
#       print('Object initialized')

    def accept(self,r,c,make):
        self.regNo=r
        self.color=c
        self.make=make
        print('values accepted')

    def display(self,a):
        print("Vehicle is",self.make)

    def __str__(self):
        return str(self.__dict__)
```

In [4]:

```python
v=Vehicle()
print(v)
v.accept(34566,"Black","Honda")
print(v)
v.display("Bus")
```

```
{}
values accepted
{'regNo': 34566, 'color': 'Black', 'make': 'Honda'}
Vehicle is Honda
```

In [9]:

```python
class Vehicle:
    serviceTax=17.54
    def __init__(self,*args):
        if(len(args)==0):
            self.color="White"
            self.make="Maruti"
            print('Object initialized')

        elif(len(args)==3):
            self.regNo=args[0]
            self.color=args[1]
            self.make=args[2]
            print('values initialized')

    def display(self,a):
        Vehicle.dealer="True Value"
        print("Vehicle make is",self.make)
        print("Vehicle Service Tax",Vehicle.serviceTax)
        print("Vehicle Dealer is",Vehicle.dealer)

    def __str__(self):
        return str(self.__dict__)

    @classmethod
    def drive(cls,speed):
        Vehicle.RTORules="Drive from left side only"
        print("I am drive vehicle with speed",speed)

    @staticmethod
    def changeServiceTax(percent):
        serviceTax=percent
        print("Service Tax changed to",Vehicle.serviceTax)
```

In [10]:

```python
v=Vehicle(35657,"Black","Honda")
print(v)
v.display("Bus")
v1=Vehicle()
print(v1)
v1.display("Car")

Vehicle.drive(100)
Vehicle.changeServiceTax(10.45)
```

```
values initialized
{'regNo': 35657, 'color': 'Black', 'make': 'Honda'}
Vehicle make is Honda
Vehicle Service Tax 17.54
Vehicle Dealer is True Value
Object initialized
{'color': 'White', 'make': 'Maruti'}
Vehicle make is Maruti
Vehicle Service Tax 17.54
Vehicle Dealer is True Value
I am drive vehicle with speed 100
Service Tax changed to 17.54
```

In [9]:

```python
# Create Account class
class Account:
    def __init__(self):
        self.accno=243355
        self.accname="Trupti Gadkari"
        self.address="Nagpur"
        print("Object initialized")

    def __init__(self,no,name,address):
        self.accno=no
        self.accname=name
        self.address=address
        print("Details of Account holder")

    def __str__(self):
        return str(self.__dict__)
```

In [10]:

```python
a=Account(355785,"Trupti Gadkari","Nagpur")
print(a)
```

```
Details of Account holder
{'accno': 355785, 'accname': 'Trupti Gadkari', 'address': 'Nagpur'}
```

In [7]:

```python
class Numbers:
    def operate(self,a=0,b=7,c=34,d=12):
        print("Addition is ",a+b+c+d)
```

In [8]:

```python
n=Numbers()
n.operate(3,4,4)
n.operate(3)
```

```
Addition is  23
Addition is  56
```

In [9]:

```python
from functools import reduce
class Numbers:
    def operate(self,*a):
        print("No of argument :",len(a))
        print("Addition is ",reduce(lambda x,y : x+y ,a))
```

In [10]:

```python
n=Numbers()
n.operate(3,4,4,5)
n.operate(3)
```

```
No of argument : 4
Addition is  16
No of argument : 1
Addition is  3
```

In [1]:

```python
pip install multipledispatch
```

```
Requirement already satisfied: multipledispatch in c:\users\user\anaconda3\lib\site-packages (0.6.0)
Requirement already satisfied: six in c:\users\user\anaconda3\lib\site-packages (from multipledispatch) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:

```python
from multipledispatch import dispatch
class Numbers:
    @dispatch(int,int)
    def operate(self,a,b):
        print("Addition is ",a+b)

    @dispatch(int,float,int)
    def operate(self,a,b,c):
        print("Addition is ",a+b+c)
```

In [6]:

```python
n=Numbers()
n.operate(3,4)
n.operate(3,0.5,6)
```

```
Addition is  7
Addition is  9.5
```

In [18]:

```python
# Inheritance
# 1) single inheritance
class Vehicle:
    def __init__(self):
        self.regNo=4566
        self.color="White"
        self.make="Maruti"
        print('In constructor of Vehicle')

    def showVehicle(self):
        Vehicle.dealer="True Value"
        print("Vehicle Reg is",self.regNo)
        print("Vehicle color is",self.color)
        print("Vehicle make is",self.make)

    def __str__(self):
        return str(self.__dict__)
```

In [19]:

```python
class TwoWheeler(Vehicle):
    def __init__(self,*a):
        if(len(a)==0):
            self.average=45
            super().__init__()
            print('In constructor of TwoWheeler')
        elif(len(a)==4):
            super().__init__(a[1],a[2],a[3])
            print('In constructor of TwoWheeler')

    def __str__(self):
        return str(self.__dict__)

    def display(self):
        print("Average of Two Wheeler is :",self.average)
```

In [20]:

```python
t=TwoWheeler()
t.display()
t.showVehicle()
print(t)
print('______________')
t1=TwoWheeler()
```

```
In constructor of Vehicle
In constructor of TwoWheeler
Average of Two Wheeler is : 45
Vehicle Reg is 4566
Vehicle color is White
Vehicle make is Maruti
{'average': 45, 'regNo': 4566, 'color': 'White', 'make': 'Maruti'}
______________
In constructor of Vehicle
In constructor of TwoWheeler
```

In [29]:

```python
#2) Multilevel
class Bike(TwoWheeler):
    def __init__(self,*a):
        if(len(a)==0):
            self.cost=100000
            super().__init__()
            print('In constructor of Bike')
        elif (len(a)==5):
            self.cost=a[0]
            super().__init__(a[1],a[2],a[3],a[4])
            print('In constructor of Bike')

    def __str__(self):
        return str(self.__dict__)

    def showBikeDetails(self):
        print("Cost of Bike is :",self.cost)
```

In [30]:

```python
b1=Bike(90000,85,45366,"Bike","Honda Shine")
print(b1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_5816/2575736176.py in <module>
----> 1 b1=Bike(90000,85,45366,"Bike","Honda Shine")
      2 print(b1)

~\AppData\Local\Temp/ipykernel_5816/3111763693.py in __init__(self, *a)
      8         elif (len(a)==5):
      9             self.cost=a[0]
---> 10             super().__init__(a[1],a[2],a[3],a[4])
     11             print('In constructor of Bike')
     12

~\AppData\Local\Temp/ipykernel_5816/3503442555.py in __init__(self, *a)
      6             print('In constructor of TwoWheeler')
      7         elif(len(a)==4):
----> 8             super().__init__(a[1],a[2],a[3])
      9             print('In constructor of TwoWheeler')
     10

TypeError: __init__() takes 1 positional argument but 4 were given
```

In [57]:

```python
#Overriding
class A:
    def show(Self):
        print("Show from A")

class B(A):
    def show(Self):
        super().show()
        print("Show from B")

class C(A):
    def show(Self):
        #super().show()
        print("Show from C")

class D(C,B):
    def show(Self):
        super().show()
        print("Show from D")
```

In [58]:

```python
# b=B()
# b.show()
# c=C()
# c.show()

d=D()
d.show()
```

```
Show from C
Show from D
```

In [74]:

```python
from abc import ABC,abstractmethod
class Shape(ABC):
    def __init__(self):
        print("Shape- abstract class constructor")
    def fillcolor(self,c):
        print("Color filled is",c)
    @abstractmethod
    def describeShape(self):
        pass
```

In [75]:

```python
# s=Shape() not possible
class Rectangle(Shape):
    def __init__(self):
        super().__init__();
        print("Rectangle constructor")
    def describeShape(self):
        print("It is rectangle with 4 sides")
```

In [76]:

```python
r=Rectangle()
r.describeShape()
r.fillcolor("Blue")
```

```
Shape- abstract class constructor
Rectangle constructor
It is rectangle with 4 sides
Color filled is Blue
```

In [23]:

```python
# Duck Typing
class Television:
    def switchON(self):
        print("Television is switched ON")
    def increaseVolume(self):
        print("Increasing volume of TV")

class MusicSystem:
    def switchON(self):
        print("MusicSystem is switched ON")
    def increaseVolume(self):
        print("Increasing volume of MusicSystem")

class ElectronicsDevice:
    def useDevice(self,homeappliance):
        homeappliance.switchON()
        homeappliance.increaseVolume()
```

In [24]:

```python
tv=Television()
m=MusicSystem()
ed=ElectronicsDevice()
ed.useDevice(tv)
ed.useDevice(m)
```

```
Television is switched ON
Increasing volume of TV
MusicSystem is switched ON
Increasing volume of MusicSystem
```

In [51]:

```python
# Monkey patching
class Mywork:
    def perform(self):
        print("It performing some task as per protocol steps")
```

In [30]:

```python
m=Mywork()
m.perform()
print(type(m.perform))
```

```
It performing some task as per protocol steps
<class 'method'>
```

In [36]:

```python
def doTask():
    pass
```

In [37]:

```python
print(type(doTask))
```

```
<class 'function'>
```

In [48]:

```python
def doTask():
    print("I am performing in funny way")
```

In [49]:

```python
doTask()
```

```
I am performing in funny way
```

In [ ]:

```python
# Assignment
```

In [ ]:

```python
'''1.Define a class Teacher described as below:-
Instance Variables:
   Name, Address, Phone, Subject Specialization, Monthly_Salary, Income Tax.
Instance methods   :
 Write constructor
 To accept the details of a teacher including the monthlysalary.
 To display the details of the teacher.
 To compute the annual Income Tax as 5% of the annual salary above Rs.1,75,000/-.
'''
```

In [13]:

```python
class Teacher:
    def __init__(self):
        self.name="Trupti"
        self.address="Nagpur"
        self.phone=7654678766
        self.subject="CA"
        self.special="Computer"
        self.msalary=60000
        self.itax=100

    def accept(self,n,a,p,s,sp,m,i):
        self.name=n
        self.address=a
        self.phone=p
        self.subject=s
        self.special=sp
        self.msalary=m
        self.itax=i

    def display(self):
        print("Name :",self.name)
        print("Address :",self.address)
        print("Phone :",self.phone)
        print("Subject :",self.subject)
        print("Specialization :",self.special)
        print("monthly salary :",self.msalary)
        print("Income Tax :",self.itax)

    def __str__(self):
        return str(self.__dict__)

    def computetax(self):
        annual_income=self.msalary*12
        print("Annual Income of",self.name,"is",annual_income)
        if(annual_income>245000):
            total_tax=annual_income*0.05
            print("Total Tax :",total_tax)
        else:
            print("Tax is not applicable for teacher whose salary is less than 245000")
```

In [14]:

```python
t=Teacher()
print(t)
print("______________________")
t.accept("Trupti","Nagpur",7654678766,"python","CSE",400000,1000)
t.display()
print("______________________")
t.computetax()
```

```
{'name': 'Trupti', 'address': 'Nagpur', 'phone': 7654678766, 'subject': 'CA', 'special': 'Computer', 'msalary': 60000, 'itax': 100}
______________________
Name : Trupti
Address : Nagpur
Phone : 7654678766
Subject : python
Specialization : CSE
monthly salary : 400000
Income Tax : 1000
______________________
Annual Income of Trupti is 4800000
Total Tax : 240000.0
```

In [ ]:

```python
'''2.Define a class ‘FoodWorld’ having the following description –
Instance Variables:
vegName : Name of the vegetable bought (string)
vegQty :    Quantity of vegetable bought (int)
vegPrice :  Price per unit of the vegetable (int)
bill : Total bill
Instance methods:
 write Constructor
 void displayData ( ):   to display the details in different line formatted as below example:
 calcBill ( ) :      to calculate the amount to be paid by the customer for the vegetable
'''
```

In [4]:

```python
class FoodWorld:
    def __init__(self,veg,qty,price):
        self.veg=veg
        self.qty=qty
        self.price=price

    def display(self):
        print("Veg dish :",self.veg)
        print("Quantity :",self.qty)
        print("Price :",self.price)

    def calcBill(self):
        totalbill=(self.qty*self.price)
        print("Total bill :",totalbill)
```

In [6]:

```python
v=FoodWorld('Panner',4,350)
v.display()
v.calcBill()
print("_________________")
v1=FoodWorld('Pizza',6,560)
v1.display()
v1.calcBill()
```

```
Veg dish : Panner
Quantity : 4
Price : 350
Total bill : 1400
_________________
Veg dish : Pizza
Quantity : 6
Price : 560
Total bill : 3360
```

In [ ]: