Assignment No.16

#Create a class Book with members as bid,bname,price and author.Add following methods:

#a. Constructor (Support both parameterized and parameterless)

#b. Destructor

#c. ShowBook

#d. Add static variable count and also maintain count of objects created.

```
class Book:
    total = 0
    def __init__(self, bid, bname, price, author):
    #def __init__(self, bid=103, bname='Dnyaneshwari', price=300, author='Dnyaneshwar
Kulkarni'):
        Book.total +=1
        self.bid = bid
        self.bname = bname
        self.price = price
        self.author = author
    def showData(self):
        return f'Book
id:{self.bid}\nBookName:{self.bname}\nPrice:{self.price}\nAuthor:{self.author}'
    def totalCount():
        print("Total Count of all Book:", Book.total)
b1 = Book(101, 'Bhagwat Geeta', 500, 'Vyas')
print(b1.showData())
print("-----------------------------------")
b2 = Book(102, 'Shyamchi Aai', 300, 'Sane Guruji')
print(b2.showData())
print("-----------------------------------")
#b3 = Book()
```

```
#print(b3.showData())

#print("-----------------------------------")

Book.totalCount()
```

#Create a class Product with members as pid,pname,price and quantity .Add following methods:

#e. Constructor (Support both parameterized and parameterless)

#f. Destructor

#g. ShowProduct

#h. Add static member discount.

#i. Provide methods for applying discount on price of product.

```
class Product:

    discount = 10

    def __init__(self, pid, pname, price, quantity):

    #def __init__(self, pid=103, pname='Pen', price=10, quantity=10):

        self.pid = pid

        self.pname = pname

        self.price = price

        self.quantity = quantity

    def showProduct(self):

        discount_price = Product.apply_discount(self.price)

        return f'Product id:{self.pid}\nProduct
Name:{self.pname}\nPrice:{self.price}\nQuantity:{self.quantity}\nDiscount:{Product.discount
}\nFinal price of product:{discount_price}'

    @staticmethod

    def apply_discount(price):

        return price - (price * Product.discount / 100)

    def __del__(self):

        print("Destructor method called")

p1 = Product(101, 'Book', 200, 5)
```

```
print(p1.showProduct())
```

#Create a class Shirt with members as sid,sname,type(formal etc), price and size(small,large etc) .Add following methods:

#j. Constructor (Support both parameterized and parameterless)

#k. Destructor

#l. ShowShirt

#m. For each size of shirt price should change by 10%.

#(eg. If 1000 is price then small price = 1000, medium = 1100,large=1200 and xlarge=1300) Use static concept.

```
class Shirt:
    m_charge = 0.1
    l_charge = 0.2
    x_charge = 0.3
    def __init__(self, sid, sname, type, price, size):
    #def __init__(self, sid=103, sname='Jocky', type='formal', price=250, size='small'):
        self.sid = sid
        self.sname = sname
        self.type = type
        self.size = size
        if(self.size == 'small'):
            self.price = price
        elif(self.size == 'medium'):
            self.price = price + (price * Shirt.m_charge)
        elif(self.size == 'large'):
            self.price = price + (price * Shirt.l_charge)
        elif(self.size == 'xlarge'):
            self.price = price + (price * Shirt.x_charge)
    def showShirt(self):
        print("Shirt ID:", self.sid)
        print("Shirt Name:", self.sname)
        print("Type:", self.type)
        print("Price:", self.price)
        print("Size:", self.size)
        print("Discount:", self.price)
        print('---------------------------')
    def __del__(self):
        print("Destructor method called")
s1 = Shirt(101, 'Cottonking', 'formal', 1000, 'large')
s2 = Shirt(102, 'Jocky', 'formal', 1000, 'small')
s1.showShirt()
s2.showShirt()
```