Assignment No:16

Q.1.#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_count=0
   def __init__(self,bid,bname,price,author):
      Book.total_count+=1
      self.book_id=bid
      self.book_nm=bname
      self.price=price
      self.author=author

   def showBook(self):
      return f'Book id:{self.book_id}\nBook Name:{self.book_nm}\nBook
Price:{self.price}\nAuthor:{self.author}'

   @staticmethod
   def count():
      print("Count:",Book.total_count)


   def __del__(self):
      print("Destructor is called:")

b1=Book(101,"Wings of Fire", 500,"Abdul Kalam")
print(b1.showBook())
print("#################")
#b2=Book(102,"Man mai hai vishwas",300,"Vishwas Nangre Patil")
#print(b2.showBook())
#print("#################")
b3=Book(103,"Sham's Mother",350,"Sane Guruji")
print(b3.showBook())

Book.count()
```

Q.2.#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. ShowBook
#h. Add static member discount.
#i. Provide methods for applying discount on price of product.

```
class Product:
   discount=15

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

      self.prod_id=pid
      self.prod_nm=pname
```

```
        self.price=price
        self.quantity=quantity

    def showProduct(self):
        discount_price = Product.apply_discount(self.price)
        return f'Product id:{self.prod_id}\nProduct
Name:{self.prod_nm}\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 is called:")


p1=Product(201,"SchoolBag", 500,1)
print(p1.showProduct())
print("########################")
p2=Product(202,"Waterbottle",200,1)
print(p2.showProduct())
print("########################")
p3=Product(203,"Notebooks",1000,5)
print(p3.showProduct())
```

Q.3.#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. ShowBook
#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):
        self.shirt_id=sid
        self.shirt_nm=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.shirt_id)
```

```
        print("Shirt Name:",self.shirt_nm)
        print("Shirt Type:",self.type)
        print("Price:",self.price)
        print("size:",self.size)
        print("Discount:",self.price)
        print("############################")



    def __del__(self):
        print("Destructor is called:")

s1=Shirt(301,"t-shirt","V-neck", 2000,"medium")
s2=Shirt(302,"Denim shirt","casual",800,"small")
s3=Shirt(303,"Line shirt","Casual",1000,"large")

#s1.showShirt()
s2.showShirt()
s3.showShirt()
```