/\*

Since java,.net and some other coding language doesn't uphold various legacy if there should be an occurrence of class, yet by utilizing it can accomplish different legacy.So the inquiry emerges when we have a theoretical subject.The explanation is, theoretical classes might contain faster factors, while factors in the interface are conclusive, public, and static.\*/

interface OrderItems{

//Declare of abstract methods nothing but method with out body

void changeItem(int newItem);

void ApplyDiscount(int increment);

void AddToCart(int decrement);

}

class ClientOrders implements OrderItems{

int discount;

int Item;

//To change order item from cart

@Override

public void changeItem(int newItem){

Item=newItem;

}

//To increase order item discount

@Override

public void ApplyDiscount(int increment){

discount=discount+increment;

}

//To decrease the discount of order item based on value changes

@Override

public void AddToCart(int decrement){

discount=discount-decrement;

}

//To display order item's in cart level

public void ShowCart{

println("display :"+discount+"item:"+Item)

}

}

class PlaceOrderItems{

//The return sort of the technique is void no compelling reason to return any qualities

public static void main(object[] argments){

//all ways object will create for derived class method

//with the help of child refrence calling parent methods

//To perform sum oprations

ClientOrders ClientOrders=new ClientOrders;

ClientOrders.changeItem(3);

ClientOrders.ApplyDiscount(10);

ClientOrders.AddToCart(6);

Order.ShowCart();

// creating instance of the Order.

Order Order = new Order();

Order.changeItem(6);

Order.ApplyDiscount(9);

Order.AddToCart(4);

println("customerOrder status :");

Order.ShowCart();

}

}