Create a Customer,Order,Lineitem and Product OOAD model and display all details of customer.

Step 1:Create a project CustomerCartLab application

Step2:Create a Product class

|  |
| --- |
| internal class Product  {  public int Id { get; set; }  public string Name { get; set; }  public decimal Price { get; set; }  public decimal DiscountPercentage { get; set; }  public decimal PriceAfterDiscount()  {  return Price \* (1 - (DiscountPercentage / 100));  }  } |

Step3:Create a LineItem class

|  |
| --- |
| internal class LineItem  {  public int Id { get; set; }  public Product Product { get; set; }  public int Quantity { get; set; }  public decimal CalculateItemCost()  {  return Product.PriceAfterDiscount() \* Quantity;  }  } |

Step4:Create a Order class

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace CustomerShoppingRefactoredApp.Model  {  internal class Order  {  public string Name { get; set; }  public int Id { get; set; }  public int Count { get { return LineItems.Count; } }  public DateTime CreatedDate { get; set; }  public List<LineItem> LineItems { get; set; }  public Order()  {  LineItems = new List<LineItem>();  }  public void Add(LineItem lineItem)  {  bool existingProduct = false;  for (int i = 0; i < LineItems.Count; i++)  {  if (LineItems[i].Product.Id == lineItem.Product.Id)  {  LineItems[i].Quantity += lineItem.Quantity;  existingProduct = true;  return;  }  }  if (!existingProduct)  {  LineItems.Add(lineItem);  }  }  public decimal CheckOutPrice()  {  decimal total = 0;  foreach (var item in LineItems)  {  total += item.CalculateItemCost();  }  return total;  }  }  } |

Step 5:Create a Customer class

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace CustomerShoppingRefactoredApp.Model  {  internal class Customer  {  public int Id { get; set; }  public string Name { get; set; }  public List<Order> Orders { get; set; }  public Customer()  {  Orders = new List<Order>();  }  public void AddOrder(Order order)  {  Orders.Add(order);  }  }  } |

Step 6: Test the program by creating each object

|  |
| --- |
| internal class Program  {  static void Main(string[] args)  {  Product book = new Product { Id = 1, Name = "Book", Price = 50, DiscountPercentage = 10 };  Product pen = new Product { Id = 2, Name = "Pen", Price = 10, DiscountPercentage = 5 };  Product chocolate = new Product { Id = 3, Name = "Chocolate", Price = 20, DiscountPercentage = 2 };  LineItem item1 = new LineItem { Id = 101, Quantity = 2, Product = book };  LineItem item2 = new LineItem { Id = 102, Quantity = 2, Product = pen };  LineItem item3 = new LineItem { Id = 103, Quantity = 1, Product = chocolate };  Order order1 = new Order { Id = 1, Name = "Dmart Order", CreatedDate = DateTime.Now };  order1.Add(item1);  order1.Add(item2);  order1.Add(item1);  order1.Add(item2);  order1.Add(item3);  Customer c = new Customer { Id = 1, Name = "John" };  c.Orders.Add(order1);  PrintDetails(c);  }  private static void PrintDetails(Customer c)  {  Console.WriteLine("Printitn order details for customer "+c.Name);  Console.WriteLine("no of order are "+c.Orders.Count);  foreach (Order order in c.Orders)  {  Console.WriteLine( "order id:"+order.Id +" date is "+order.CreatedDate);  PrintDetails(order.LineItems);  }  }  private static void PrintDetails(List<LineItem> lineItems)  {  foreach(LineItem lineItem in lineItems) {  Console.WriteLine( "Lineitem is "+lineItem.Id+ "name of product "+lineItem.Product.Name+ " cost"+lineItem.CalculateItemCost());  }  }  } |

//