using System;

// Encapsulation: Product Class

class Product

{

private int productID;

private string name;

private decimal price;

private int stockQuantity;

public int ProductID

{

get { return productID; }

set { productID = value; }

}

public string Name

{

get { return name; }

set { name = value; }

}

public decimal Price

{

get { return price; }

set

{

if (value < 0)

throw new ArgumentException("Price cannot be negative.");

price = value;

}

}

public int StockQuantity

{

get { return stockQuantity; }

set

{

if (value < 0)

throw new ArgumentException("Stock quantity cannot be negative.");

stockQuantity = value;

}

}

}

// Inheritance: ElectronicProduct Class

class ElectronicProduct : Product

{

public int WarrantyPeriod { get; set; } // In months

public string Brand { get; set; }

}

// Inheritance: ClothingProduct Class

class ClothingProduct : Product

{

public string Size { get; set; }

public string Material { get; set; }

}

// Abstraction: Order Abstract Class

abstract class Order

{

public int OrderID { get; set; }

public string CustomerName { get; set; }

public DateTime OrderDate { get; set; }

public abstract decimal CalculateTotal();

}

// Concrete Class: OnlineOrder

class OnlineOrder : Order

{

private Product[] products;

private int currentIndex = 0;

public OnlineOrder(int maxProducts)

{

products = new Product[maxProducts];

}

public void AddProduct(Product product)

{

if (currentIndex >= products.Length)

{

Console.WriteLine("Cannot add more products. The order is full.");

return;

}

products[currentIndex++] = product;

}

public override decimal CalculateTotal()

{

decimal total = 0;

for (int i = 0; i < currentIndex; i++)

{

total += products[i].Price;

}

return total;

}

public void DisplayOrderDetails()

{

Console.WriteLine("Products in the order:");

for (int i = 0; i < currentIndex; i++)

{

var product = products[i];

if (product is ElectronicProduct electronic)

{

Console.WriteLine($"{electronic.Name}: ${electronic.Price} (Brand: {electronic.Brand}, Warranty: {electronic.WarrantyPeriod} years)");

}

else if (product is ClothingProduct clothing)

{

Console.WriteLine($"{clothing.Name}: ${clothing.Price} (Size: {clothing.Size}, Material: {clothing.Material})");

}

else

{

Console.WriteLine($"{product.Name}: ${product.Price}");

}

}

Console.WriteLine($"Order Total: ${CalculateTotal()}");

}

}

// Main Program

class Program

{

static void Main(string[] args)

{

// Create products

var laptop = new ElectronicProduct

{

ProductID = 1,

Name = "Laptop",

Price = 1000,

StockQuantity = 10,

WarrantyPeriod = 2,

Brand = "Dell"

};

var tshirt = new ClothingProduct

{

ProductID = 2,

Name = "T-Shirt",

Price = 20,

StockQuantity = 50,

Size = "M",

Material = "Cotton"

};

// Create an order with a maximum of 5 products

var order = new OnlineOrder(5)

{

OrderID = 101,

CustomerName = "John Doe",

OrderDate = DateTime.Now,

};

// Add products to the order

order.AddProduct(laptop);

order.AddProduct(tshirt);

// Display order details

order.DisplayOrderDetails();

}

}