Program.cs | Medicine.cs | MedicineUtility.cs

OK

```
namespace PharmaDetail //DO NOT change the namespace name
{
    public class Program //DO NOT change the class name
    {
        public static void Main(string[] args)  //DO NOT change the method signature
        {
            AddMedicineTest addMedicineObj = new AddMedicineTest(); // Do not remove this line , Implementation for Add functionlity is given in backend
            addMedicineObj.AddMedicineDetails("sample.xml");  // You can display the data from this sample file

            //Implement your code here
            MedicineUtility utility= new MedicineUtility();
            Medicine medicine = utility.GetAllMedicineDetails("sample.xml");
            if(medicine!=null){
            Console.WriteLine("Id\tName\tCost\tQuantity\tWeight");
            Console.WriteLine($"{medicine.Id}\t{medicine.Name}\t{medicine.Cost}\t{medicine.QuantityInStock}\t{medicine.Weight}");
            }
            else
            {
                Console.WriteLine("Failed to read medicine details from the file.");
            }
        }
    }
}
```


Program.cs | Medicine.cs | MedicineUtility.cs

```
using System.Xml.Serialization;
namespace PharmaDetail //DO NOT change the namespace name
{
    [Serializable]
    public class Medicine //Make the class serializable. DO NOT change the class name
    {
        //Implement your code here

        public int Id{get;set;}

        public string Name{get;set;}

        public double Cost{get;set;}

        public int QuantityInStock{get;set;}

        public float Weight{get;set;}
    }
}
```


Program.cs | Medicine.cs | MedicineUtility.cs

```
using System.Xml.Serialization;

namespace PharmaDetail //DO NOT change the namespace name
{
    public class MedicineUtility //DO NOT change the class name
    {
        //Implement your code here
        public Medicine GetAllMedicineDetails(string fileName)
        {
            try
            {
                XmlSerializer serializer= new XmlSerializer(typeof(Medicine));
                using (FileStream fileStream= new FileStream(fileName, FileMode.Open))
                {
                    return (Medicine)serializer.Deserialize(fileStream);
                }
            }
            catch(Exception ex)
            {
                Console.WriteLine("Error: "+ ex.Message);
                return null;
            }
        }
    }
}
```

Program.cs | Event.cs | EventUtility.cs

```
namespace EventManagement //DO NOT change the namespace name
{
    public class Program //DO NOT change the class name
    {
        public static void Main(string[] args)  //DO NOT change the method signature
        {
            //Implement your code here
            Event eventObj= new Event();

            Console.WriteLine("Enter Event's Name");
            eventObj.Name= Console.ReadLine();

            Console.WriteLine("Enter Event's Description");
            eventObj.Description= Console.ReadLine();

            Console.WriteLine("Enter Event's Type");
            eventObj.Type= Console.ReadLine();

            Console.WriteLine("Enter the name of the person who Created the event");
            eventObj.CreatedBy= Console.ReadLine();

            Console.WriteLine("Enter the location");
            eventObj.Location= Console.ReadLine();

            Console.WriteLine("Enter the Event Starting Date (dd/MM/yyyy)");
            eventObj.StartDate= DateTime.ParseExact(Console.ReadLine(),"dd/MM/yyyy",null);
            Console.WriteLine("Enter the Event Ending Date (dd/MM/yyyy)");
            eventObj.EndDate= DateTime.ParseExact(Console.ReadLine(),"dd/MM/yyyy",null);

            Console.WriteLine("Enter the FileName in '.xml' format");
            string fileName= Console.ReadLine();
            EventUtility eventUtility = new EventUtility();
            eventUtility.AddEventDetails(eventObj, fileName);
        }
    }
}
```

Program.cs | Event.cs | EventUtility.cs

```
using System.IO;
using System.Xml.Serialization;
namespace EventManagement //DO NOT change the namespace name
{
    [Serializable]
    public class Event //Make the class serializable. DO NOT change the class name
    {
        //Implement your code here
        public string Name {get; set;}
        public string Description {get; set;}
        public string Type {get; set;}
        public string CreatedBy {get; set;}
        public string Location {get; set;}
        //[XmlElement(DataType="date")]
        public DateTime StartDate {get; set;}
        //[XmlElement(DataType="date")]
        public DateTime EndDate {get; set;}
    }
}
```

Program.cs | Event.cs | EventUtility.cs

```
using System.Runtime.Serialization;
using System.Xml.Serialization;

namespace EventManagement //DO NOT change the namespace name
{
    public class EventUtility //DO NOT change the class name
    {
        //Implement your code here
        public void AddEventDetails(Event eventObj, string fileName)
        {
            try{
                XmlSerializer serializer= new XmlSerializer(typeof(Event));
                using (StreamWriter writer= new StreamWriter(fileName))
                {
                    serializer.Serialize(writer, eventObj);
                }
                Console.WriteLine("Event Details Added Successfully");
            }
            catch
            {
                Console.WriteLine("Event Details Cannot be added");
            }

        }
    }
}
```

Program.cs | BillDetails.cs | EBService.cs

```
namespace ElectricityBill //DO NOT change the namespace name
{
    public class Program //DO NOT change the class name
    {
        public static void Main(string[] args)  //DO NOT change the method signature
        {
            AddEBDetailsTest testObj = new AddEBDetailsTest(); // Do not remove this line , Implementation for Add func
            testObj.AddEBDetails("sample.txt"); // You can display the data from this sample file

            //Implement your code here
            EBService ebService = new EBService();
            BillDetails billDetails= ebService.ReadEBDetails("sample.txt");
            if(billDetails!=null)
            {
                Console.WriteLine("\nConsumerNumber");
                Console.WriteLine("ConsumerName");
                Console.WriteLine("UnitsConsumed");
                Console.WriteLine("BillAmount");
                Console.WriteLine("\n{0}\n{1}\n{2}\n{3}",
                billDetails.ConsumerNumber,
                billDetails.ConsumerName,
                billDetails.UnitsConsumed,
                billDetails.BillAmount);
            }
        }
    }
}
```

Program.cs | BillDetails.cs | EBService.cs

```
namespace ElectricityBill //DO NOT change the namespace name
{
    [Serializable]
    public class BillDetails //Make the class serializable. DO NOT change the class name
    {
        //Implement your code here
        public string ConsumerNumber{get; set;}
        public string ConsumerName{get; set;}
        public int UnitsConsumed{get;set;}
        public double BillAmount{get; set;}
    }
}
```

Program.cs | BillDetails.cs | EBService.cs

```cs
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;

namespace ElectricityBill //DO NOT change the namespace name
{
    public class EBService //DO NOT change the class name
    {
        //Implement your code here
        public BillDetails ReadEBDetails(string fileName)
        {
            try{
                using (FileStream fileStream = new FileStream(fileName, FileMode.Open, FileAccess.Read))
                {
                    IFormatter formatter= new BinaryFormatter();
                    return (BillDetails)formatter.Deserialize(fileStream);
                }
            }
            catch(Exception ex)
            {
                Console.WriteLine("Error: "+ ex.Message);
                return null;
            }
        }
    }
}

```

Program.cs | Shipment.cs | ShipmentService.cs

```cs
namespace CargoManagementSystem //DO NOT change the namespace name
{
    public class Program //DO NOT change the class name
    {
        public static void Main(string[] args)  //DO NOT change the method signature
        {
            //Implement your code here
            Shipment shipment= new Shipment();
            Console.WriteLine("Enter the shipment id");
            shipment.ShipmentId= int.Parse(Console.ReadLine());

             Console.WriteLine("Enter the customer id");
            shipment.CustomerId= Console.ReadLine();

            Console.WriteLine("Enter start location");
            shipment.StartLocation= Console.ReadLine();

            Console.WriteLine("Enter end location");
            shipment.EndLocation= Console.ReadLine();

            Console.WriteLine("Enter expected start date");
            shipment.ExpectedStartDate= DateTime.Parse(Console.ReadLine());

             Console.WriteLine("Enter actual start date");
            shipment.ActualStartDate= DateTime.Parse(Console.ReadLine());

            Console.WriteLine("Enter expected end date");
            shipment.ExpectedEndDate= DateTime.Parse(Console.ReadLine());

             Console.WriteLine("Enter actual end date");
            shipment.ActualEndDate= DateTime.Parse(Console.ReadLine());

            Console.WriteLine("Enter file name");
            string fileName= Console.ReadLine();
            ShipmentService service= new ShipmentService();
            service.AddShipmentDetails(shipment, fileName);
        }
    }
}

```

Program.cs | Shipment.cs | ShipmentService.cs

```
namespace CargoManagementSystem //DO NOT change the namespace name
{
    [Serializable]
    public class Shipment //Make the class serializable. DO NOT change the class
    {
        //Implement your code here
        public int ShipmentId{get; set;}
        public string CustomerId{get; set;}
        public string StartLocation{get; set;}
        public string EndLocation{get; set;}
        public DateTime ExpectedStartDate{get; set;}
        public DateTime ActualStartDate{get; set;}
        public DateTime ExpectedEndDate{get; set;}
        public DateTime ActualEndDate{get; set;}
    }
}
```


Program.cs | Shipment.cs | ShipmentService.cs

```
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;

namespace CargoManagementSystem //DO NOT change the namespace name
{
    public class ShipmentService //DO NOT change the class name
    {
        //Implement your code here
        public void AddShipmentDetails(Shipment shipmentobj, string fileName)
        {
            try
            {
                using (FileStream fileStream=new FileStream(fileName, FileMode.Create))
                {
                    BinaryFormatter formatter= new BinaryFormatter();
                    formatter.Serialize(fileStream, shipmentobj);
                }
                Console.WriteLine("Shipment details added successfully");
            }
            catch(Exception ex)
            {
                Console.WriteLine("Error: "+ ex.Message);
            }
        }
    }
}
```

Program.cs | MyFile.txt

```
namespace FileIO //DO NOT change the namespace name
{
    public class Program //DO NOT change the class name
    {
        //Implement your method here
        public void WriteOnFile(string fileName, string text)
        {
            File.WriteAllText(fileName, text);
        }
        public void AppendFile (string fileName, string text)
        {
            File.AppendAllText(fileName, " "+ text);
        }
        public string[] ReadFile(string fileName)
        {
            return File.ReadAllLines(fileName);
        }

       public static void Main(string[] args)  //DO NOT change the method signature
       {
           //Implement your code here
           Program program= new Program();
           bool exit= false;
           while(!exit)
           {
               Console.WriteLine("1. Write on file");
               Console.WriteLine("2. Append file");
               Console.WriteLine("3. Read file");
               Console.WriteLine("Enter your choice");

               if(!int.TryParse(Console.ReadLine(), out int choice))
               {
                   Console.WriteLine("Invalid input. Please enter a number between 1 and 4.");
                   continue;
               }
               Console.WriteLine("Enter the filename");
               string fileName= Console.ReadLine();
               switch(choice)
               {
                   case 1:
                   // Console.WriteLine("Enter the filename");
                   // string fileName= Console.ReadLine();
                    Console.WriteLine("Enter the text to write");
                    string text= Console.ReadLine();
                    program.WriteOnFile(fileName, text);
                    break;
                    case 2:
                   // Console.WriteLine("Enter the file name");
                   // fileName=Console.ReadLine();
                    Console.WriteLine("Enter the text to append");
                    text= Console.ReadLine();
                    program.AppendFile(fileName, text);
                    break;

                    case 3:
                   // Console.WriteLine("Enter the file to read");
                   //  fileName= Console.ReadLine();
                    string[] content= program.ReadFile(fileName);
                    Console.WriteLine(string.Join(" ", content));
                    break;

                    default:
                    Console.WriteLine("Invalid choice. Please enter a number between 1 and 3.");
                    break;
               }
           }
       }
    }
}
```