Họ và Tên: Nguyễn Thanh Kiên.

MSSV: 22110092.

Bài tập:

Xây dựng lớp sản phẩm sữa Milk với các thuộc tính sau MilkName, MilkID, ProductionDate (ngày sản xuất), ExpiredDate (ngày hết hạn), Quantity (số lượng) với các yêu cầu sau

- Các thuộc tính có các thao tác get và set tương ứng ✓
- MilkID được cấu thành như sau: MILK01032016 (MILK + Ngày sản xuất) ✓
- Viết lớp MilkMoreInfo như là một attribute cung cấp thêm thông tin Manufacturer (Nhà sản xuất), CompanyName (tên công ty phân phối sữa) cho lớp Milk ✓
- Viết một interface gồm hai thao tác là Nhập và Xuất thông tin ra màn hình. Thực hiện gọi hai thao tác này cho bất kỳ một thực thể milk nào ✓
- Viết delegate để thực hiện hai thao tác nhập và xuất này. ✓

Làm thêm:

. Validation & Error Handling ✓

. Đọc/Ghi JSON ✓

Bài làm:

1. Xây dựng lớp sản phẩm sữa Milk với các thuộc tính sau MilkName, MilkID, ProductionDate (ngày sản xuất), ExpiredDate (ngày hết hạn), Quantity (số

lượng), Các thuộc tính có các thao tác get và set tương ứng, MilkID được cấu thành như sau: MILK01032016

+ Đầu tiên, em khai các các thư viện System, tạo Namespace đầu tiên có tên MilkManagement trong đó chứa lớp Program có chứa hàm main để hiển thị ra màn hình Console, ngoài ra nó còn dùng để chứa các tham số mặc định của các thuộc tính MilkName, MilkID, ProducTionDate, ExpiredDate và Quantity

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Globalization;
using MainData;

namespace MilkManagement
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Milk milk = new Milk();
            milk.ValMilkName = "Vinamilk";
            milk.ValProductionDate = "01/03/2016";
            milk.ValExpiredDate = "20/05/2017";
            milk.ValQuantity = 123456;

            Console.WriteLine("Thong tin sua la:");
            Console.WriteLine("\t MilkID = {0}, MilkName = {1}", milk.ValMilkID, milk.ValMilkName);
            Console.WriteLine("\t ProductionDate = {0}, ExpiredDate = {1}", milk.ValProductionDate, milk.ValExpiredDate);
            Console.WriteLine("\t Quantity = {0} San Pham", milk.ValQuantity);
            Console.ReadKey();
        }
    }
}
```

+ Tiếp theo, em tạo Namespace thứ 2 – MainData với lớp Milk dùng để khai báo các thuộc tính yêu cầu và phương thức get set tương ứng với mỗi thuộc tính, trong đó có 2 phương thức thiết lập dạng chuỗi, 2 phương thức thiết lập dạng ngày tháng năm và 1 phương thức dạng giá trị số, tất cả các phương thức thiết lập này đều có tham số mặc định trong hàm main. Em sử dụng kiểu Datetime để hiển thị thông tin PublishDate trong MilkID, tuy vậy thông tin ExpiredDate không có trong MilkID, vì vậy, em cần phải sử dụng một điều kiện IF – ELSE để kiểm tra xem ExpiredDate đó có hay là không, tức giá trị của nó là NULL hay là một giá trị mặc định nào khác. Sau đó ở phương thức

get set của ExpiredDate, em lại tiếp tục dùng thêm một điều kiện IF - ELSE để kiểm tra xem nó có phải NULL hay là không.

```
namespace MainData
{
    3 references
    class Milk
    {
        private string MilkID = "MILK01032016";
        private string MilkName;
        private DateTime ProductionDate;
        private DateTime? ExpiredDate;
        private float Quantity;

        1 reference
        public Milk(string MilkName = "Not Assgined", string ProductionDate = "01/03/2016",
                    string ExpiredDate = null, float Quantity = 0)
        {
            this.MilkName = MilkName;
            this.ProductionDate = DateTime.ParseExact(ProductionDate, "dd/MM/yyyy", CultureInfo.InvariantCulture);
            if (!string.IsNullOrEmpty(ExpiredDate))
            {
                this.ExpiredDate = DateTime.ParseExact(ExpiredDate, "dd/MM/yyyy", CultureInfo.InvariantCulture);
            }
            else
            {
                this.ExpiredDate = null; // hoặc một giá trị mặc định nào đó
            }
            this.Quantity = Quantity;
            this.MilkID = String.Format("MILK{0}", this.ProductionDate.ToString("ddMMYYYY"));
        }
```

```
        1 reference
        public string ValMilkID
        {
            get { return MilkID; }
        }

        2 references
        public string ValMilkName
        {
            get { return MilkName; }
            set { MilkName = value; }
        }

        2 references
        public string ValProductionDate
        {
            get { return ProductionDate.ToString("dd/MM/yyyy"); }
            set {
                ProductionDate = DateTime.ParseExact(value, "dd/MM/yyyy", CultureInfo.InvariantCulture);
                MilkID = String.Format("MILD{0}", this.ProductionDate.ToString("ddMMyyyy"));
            }
        }
```

```
        public string ValExpiredDate
        {
            get { return ExpiredDate.HasValue ? ExpiredDate.Value.ToString("dd/MM/yyyy") : "Not Assigned"; }
            set
            {
                if (!string.IsNullOrEmpty(value))
                {
                    ExpiredDate = DateTime.ParseExact(value, "dd/MM/yyyy", CultureInfo.InvariantCulture);
                }
                else
                {
                    ExpiredDate = null;
                }
            }
        }

        2 references
        public float ValQuantity
        {
            get { return Quantity; }
            set { Quantity = value; }
        }
    }
}
```

Màn hình Console:

```
Thong tin sua la:
        MilkID = MILK01032016, MilkName = Vinamilk
        ProductionDate = 01/03/2016, ExpiredDate = 20/05/2017
        Quantity = 123456 San Pham
```

2. Viết lớp MilkMoreInfo như là một attribute cung cấp thêm thông tin Manufacturer (Nhà sản xuất), CompanyName (tên công ty phân phối sữa) cho lớp Milk

+ Em tạo Namespace thứ 3 tên là AttributeDate, trong đó em xây dựng một lớp MilkMoreInfo kế thừa từ System.Attribute, MilkMoreInfo sẽ cung cấp thông tin của Manufacturer và CompanyName. Namespace này cũng thêm phương thức khởi tạo có tham số và phương thức get & set.

```
namespace AttributeData
{
    [AttributeUsage(AttributeTargets.All, AllowMultiple = true)]
    3 references
    public class MilkMoreInfo : System.Attribute
    {
        2 references
        public string Manufacturer { get; set; }
        2 references
        public string CompanyName { get; set; }

        1 reference
        public MilkMoreInfo(string Manufacturer = "", string CompanyName = "")
        {
            this.Manufacturer = Manufacturer;
            this.CompanyName = CompanyName;
        }
    }
}
```

+ Ở Namespace MainData, em thêm dòng ghi chú vào đầu lớp Milk để cho biết thông tin của Manufacturer và Company

```
namespace MainData
{
    [MilkMoreInfo("VietNamProduction", "Vina Company")]
    4 references
    class Milk
    {
        private string MilkID = "MILK01032016";
        private string MilkName;
        private DateTime ProductionDate;
        private DateTime? ExpiredDate;
        private float Quantity;
```

+ Ở Namespace MilkManagement, em bổ sung hàm main để thêm các thông tin của Manufacturer và CompanyName và vẫn giữ nguyên các thông tin ban đầu của Milk. Ở hàm main mới này, nếu em sử dụng MilkMoreInfo minfo = (MilkMoreInfo)Attribute, nó trả về tất cả các thuộc tính đính kèm trên lớp, bao gồm các thuộc tính không phải là MilkMoreInfo. Vậy nên khi chạy chương trình nó sẽ báo lỗi, do đó, em sử dụng một câu điều kiện IF – Else if (attributes is MilkMoreInfo minfo) để kiểm tra xem thuộc tính có của MilkMoreInfo hay là không, nếu có thì sẽ xuất hiện Output Message của Manufacturer và CompanyName

```
Type type = typeof(Milk);
string OutputMessage = "Thong tin san xuat cua sua:";
foreach(Object attributes in type.GetCustomAttributes(false))
{
    //MilkMoreInfo minfo = (MilkMoreInfo)Attribute;
    if (attributes is MilkMoreInfo minfo)
    {
        OutputMessage = String.Format("\n Manufacturer = {0},\n", minfo.Manufacturer);
        OutputMessage += String.Format("\n CompanyName = {0} \n", minfo.CompanyName);

        Console.WriteLine(OutputMessage);
    }
}
```

Màn hình Console:

```
 Manufacturer = VietNamProduction,

 CompanyName = Vina Company

Thong tin sua la:
        MilkID = MILK01032016, MilkName = Vinamilk
        ProductionDate = 01/03/2016, ExpiredDate = 20/05/2017
        Quantity = 123456 San Pham
```

3. Viết một interface gồm hai thao tác là Nhập và Xuất thông tin ra màn hình. Thực hiện gọi hai thao tác này cho bất kỳ một thực thể milk nào

+ Em tạo một Interface trong Namespace MainData với hai phương thức Nhập (1 file .txt) và Xuất thông tin file txt đó bằng 1 file txt khác. Em cho lớp Milk kế thừa và sử dụng interface bằng class Milk : ImilkAction, sau đó em dùng StreamReader và StreamWriter để nhập và xuất file.

```
namespace MainData
{
    0 references
    interface IMilkAction
    {
        0 references
        int InputInfo(string filename);
        0 references
        int OutputInfo(string filename);
    }
```

```
class Milk
{
    1 reference
    public int InputInfo(string filename)
    {
        int counter = 0;
        string line;

        System.IO.StreamReader file =
            new System.IO.StreamReader(filename);
        while ((line = file.ReadLine()) != null)
        {
            string[] terms = line.Split(':');
            if (line.Contains("MilkName:")) ValMilkName = terms[1];
            if(line.Contains("ProductionDate :")) ValProductionDate = terms[1].Trim();
            if(line.Contains("ExpiredDate :")) ValExpiredDate = terms[1].Trim();
            if(line.Contains("Quantity :")) ValQuantity = float.Parse(terms[1].Trim());
            counter++;
        }
        file.Close();
        return counter;
    }
```

```
    1 reference
    public int OutputInfo(string filename)
    {
        System.IO.StreamWriter file = new System.IO.StreamWriter(filename);

        string OutputMessage = "";
        OutputMessage = String.Format("\n MilkID = {0}, \n MilkName = {1}", ValMilkID, ValMilkName);
        file.WriteLine(OutputMessage);
        OutputMessage = String.Format("\n ProductionDate = {0}, \n ExpiredDate = {1}", ValProductionDate, ValExpiredDate);
        file.WriteLine(OutputMessage);
        OutputMessage = string.Format("\n Quantity = {0} San Pham", ValQuantity);
        file.WriteLine(OutputMessage);

        file.Close();
        return 1;
    }
```

+ Trong Namespace MilkManagement, em gọi lại hai phương thức Nhập và Xuất file này, và sử dụng một điều kiện IF để kiểm tra xem dữ liệu có rỗng hay không với điều kiện numline > 0.

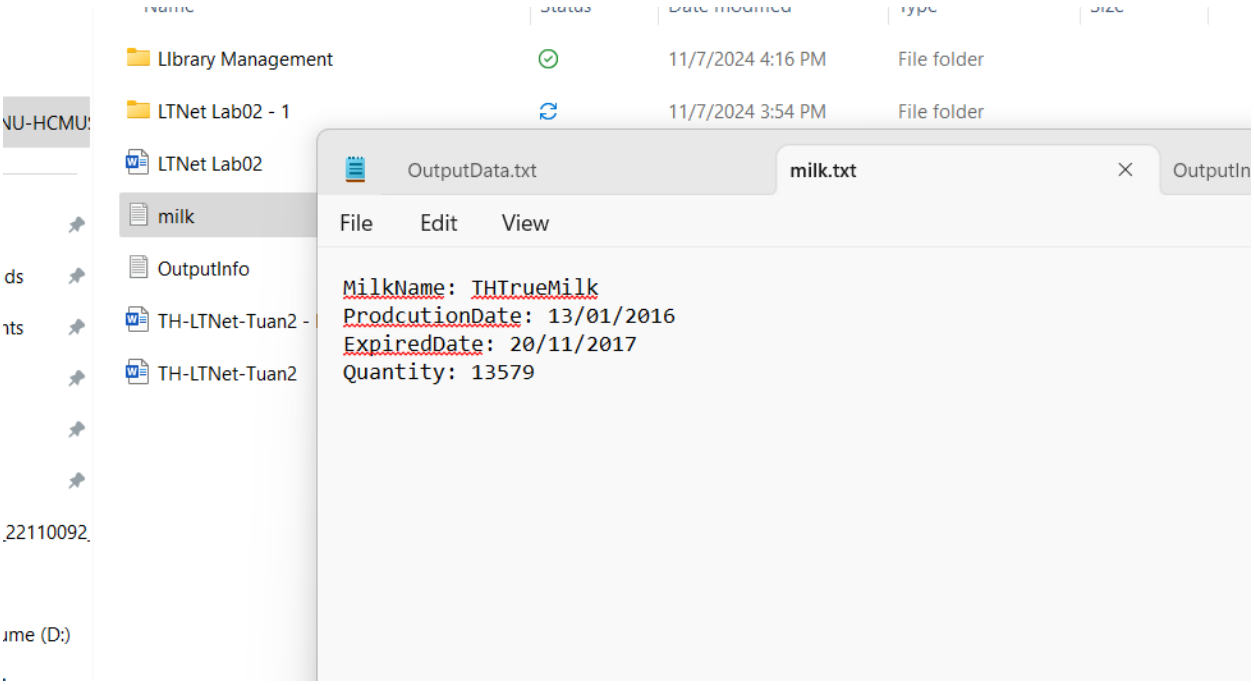
```
0 references
static void Main(string[] args)
{
    Milk milk = new Milk();
    milk.ValMilkName = "Vinamilk";
    milk.ValProductionDate = "01/03/2016";
    milk.ValExpiredDate = "20/05/2017";
    milk.ValQuantity = 123456;

    int numline = milk.InputInfo("C:\\Users\\T K\\OneDrive - VNU-HCMUS\\Desktop\\Projects ki 5\\Project .NET\\LAB 02\\milk.txt");
    if (numline > 0)
    {
        Console.WriteLine("Thong tin sua la:");
        Console.WriteLine("\t MilkID = {0}, MilkName = {1}", milk.ValMilkID, milk.ValMilkName);
        Console.WriteLine("\t ProductionDate = {0}, ExpiredDate = {1}", milk.ValProductionDate, milk.ValExpiredDate);
        Console.WriteLine("\t Quantity = {0} San Pham", milk.ValQuantity);
    }

    milk.ValQuantity = 789;
    milk.OutputInfo("C:\\Users\\T K\\OneDrive - VNU-HCMUS\\Desktop\\Projects ki 5\\Project .NET\\LAB 02\\OutputInfo.txt");
    Console.WriteLine("Write to file Successfully");

    Console.ReadKey();
}
```

Màn hình Console:

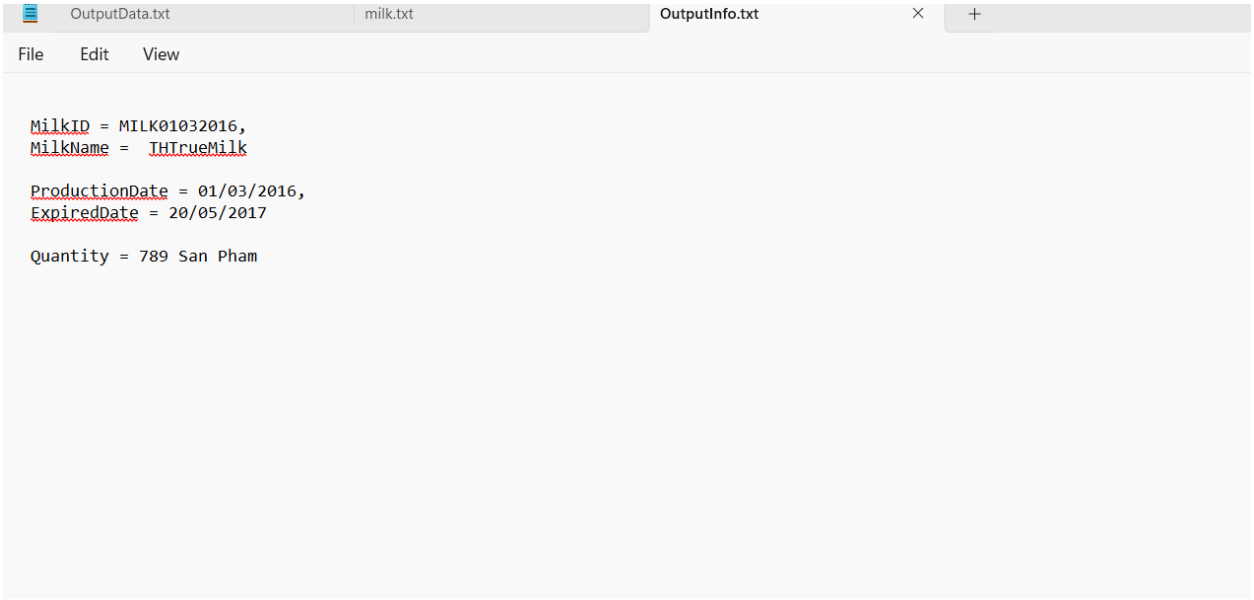
```
Thong tin sua la:
        MilkID = MILK01032016, MilkName =  TH True Milk
        ProductionDate = 01/03/2016, ExpiredDate = 20/05/2017
        Quantity = 123456 San Pham
Write to file Successfully
```

Màn hình console hiển thị thông tin sau khi đọc dữ liệu từ file input milk.txt và cập nhật một số thuộc tính trong đối tượng Milk, giá trị MilkName được cập nhật thành "THTrueMilk" sau khi đọc từ file milk.txt. Giá trị Quantity là 123456, khác với giá trị 13579 trong file input milk.txt. Đây là do thuộc tính Quantity được gán một giá trị mới 123456 trong phần code trước khi hiển thị trên màn hình console.

Input:



Output:



Sau khi đọc từ file milk.txt, thuộc tính Quantity của đối tượng Milk được thay đổi thành 789 và các thuộc tính còn lại vẫn giữ nguyên. Do đó, giá trị Quantity trong file output khác với giá trị trong file input (13579), và ExpiredDate cũng thay đổi từ 20/11/2017 sang 20/05/2017.

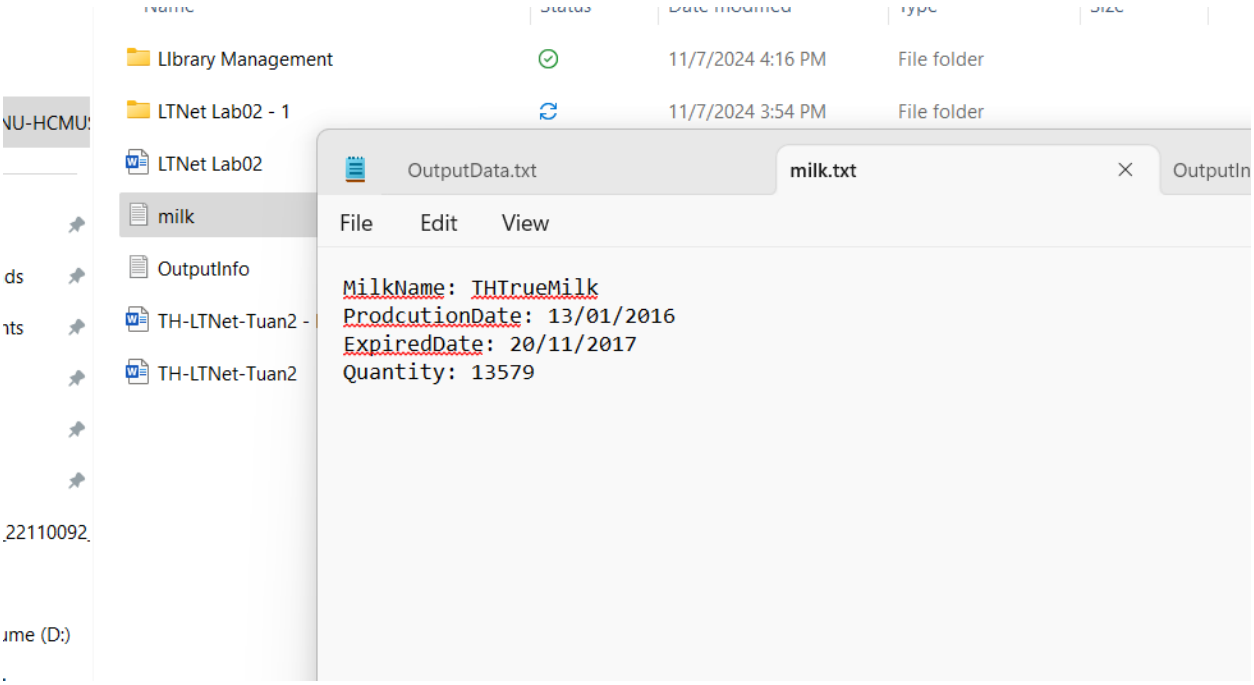
4. Viết delegate để thực hiện hai thao tác nhập và xuất này.

+ Bên ngoài Namespaces MilkManagement, em khai báo 1 delegate – con trỏ hàm MilkInputOuput trỏ đến hàm InputInfo và OutputInfo

+ Ở hàm main, em khai báo hai đối tượng action1 và action2 thuộc kiểu delegate đó và tham chiếu lần lượt đến các hàm nhập và xuất file trong lớp Milk, em dùng Invoke để yêu cầu 2 kiểu con trỏ hàm action1 và action2 thực thi 2 thao tác nhập và xuất như em mong muốn.

Màn hình Console:

Input:

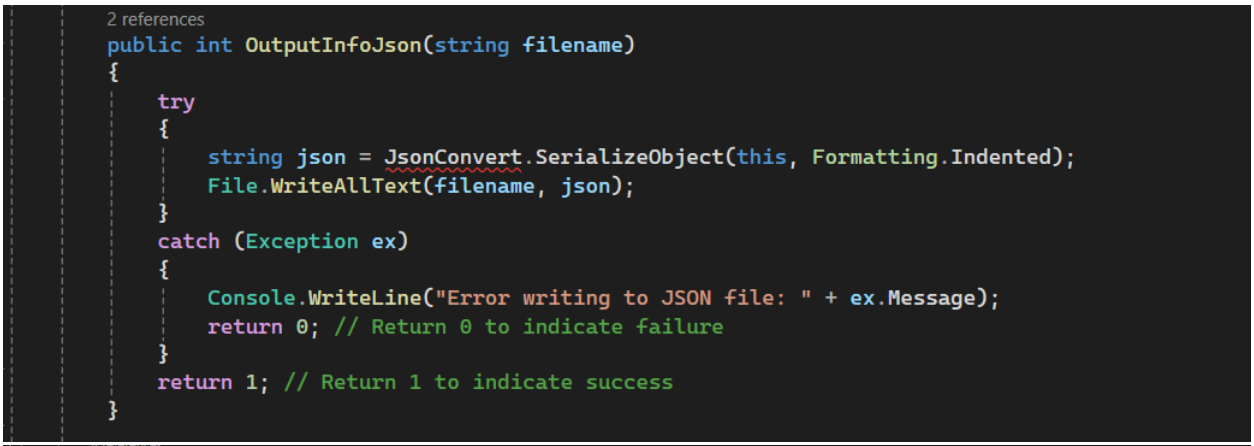


Output:

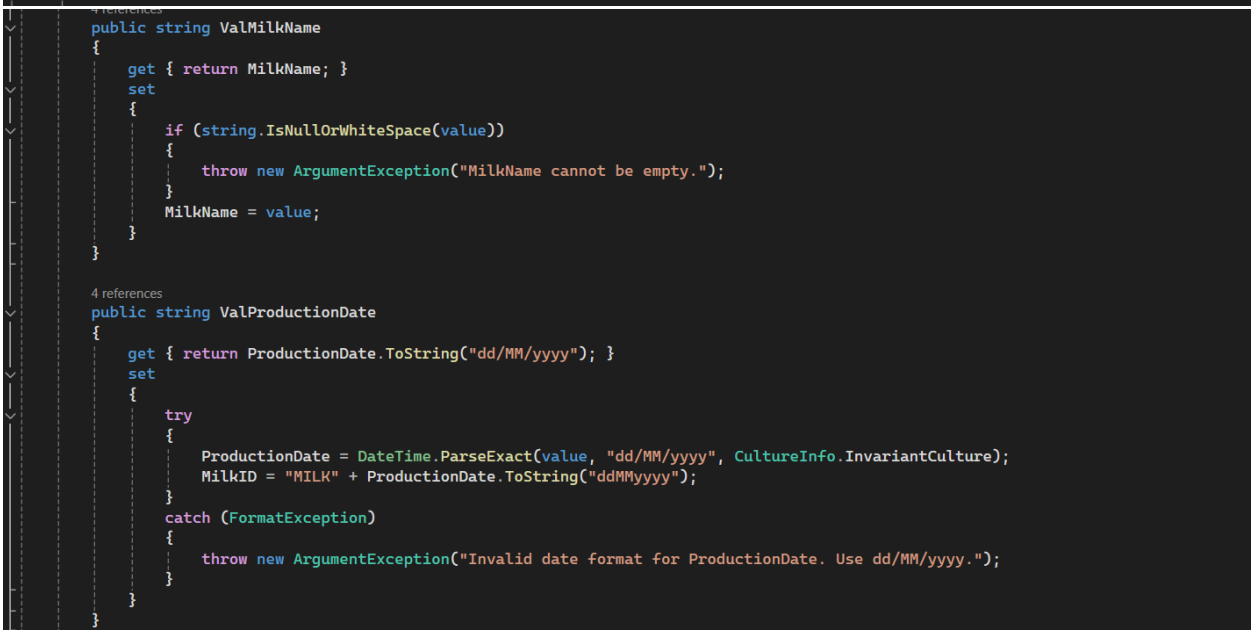
Không có sự khác biệt nào giữa 2 file OutputInfo và OutputInfoByDelegate. Ta sử dụng Delegate để tối ưu hóa code hơn, làm cho chương trình chạy nhanh hơn và tốt hơn so với việc sử dụng điều kiện IF.

Làm thêm:

1. **Validation & Error Handling**:
   Em thêm kiểm tra ArgumentException trong ValMilkName, ValProductionDate, ValExpiredDate, và ValQuantity để đảm bảo dữ liệu đầu vào hợp lệ và sử dụng try-catch trong phương thức InputInfoJson và OutputInfoJson để xử lý các lỗi khi đọc và ghi dữ liệu.



```
2 references
public int OutputInfoJson(string filename)
{
    try
    {
        string json = JsonConvert.SerializeObject(this, Formatting.Indented);
        File.WriteAllText(filename, json);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error writing to JSON file: " + ex.Message);
        return 0; // Return 0 to indicate failure
    }
    return 1; // Return 1 to indicate success
}
```



```
public string ValMilkName
{
    get { return MilkName; }
    set
    {
        if (string.IsNullOrWhiteSpace(value))
        {
            throw new ArgumentException("MilkName cannot be empty.");
        }
        MilkName = value;
    }
}

4 references
public string ValProductionDate
{
    get { return ProductionDate.ToString("dd/MM/yyyy"); }
    set
    {
        try
        {
            ProductionDate = DateTime.ParseExact(value, "dd/MM/yyyy", CultureInfo.InvariantCulture);
            MilkID = "MILK" + ProductionDate.ToString("ddMMyyyy");
        }
        catch (FormatException)
        {
            throw new ArgumentException("Invalid date format for ProductionDate. Use dd/MM/yyyy.");
        }
    }
}
```

2. **Đọc/Ghi JSON**:

Em thay đổi từ việc đọc/ghi file văn bản sang JSON bằng cách sử dụng JsonConvert.DeserializeObject và JsonConvert.SerializeObject từ thư viện Newtonsoft.Json và tạo file JSON input_milk.json để nhập dữ liệu và output_milk.json để xuất dữ liệu.

```
2 references
public int InputInfoJson(string filename)
{
    try
    {
        string json = File.ReadAllText(filename);
        Milk milkData = JsonConvert.DeserializeObject<Milk>(json);

        ValMilkName = milkData.ValMilkName;
        ValProductionDate = milkData.ValProductionDate;
        ValExpiredDate = milkData.ValExpiredDate;
        ValQuantity = milkData.ValQuantity;
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error reading from JSON file: " + ex.Message);
        return 0; // Return 0 to indicate failure
    }
    return 1; // Return 1 to indicate success
}
```

```
public int OutputInfoJson(string filename)
{
    try
    {
        string json = JsonConvert.SerializeObject(this, Formatting.Indented);
        File.WriteAllText(filename, json);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error writing to JSON file: " + ex.Message);
        return 0; // Return 0 to indicate failure
    }
    return 1; // Return 1 to indicate success
}
```