

## 3 – DAY – TASK (21-08-2024)

### 1. OpenableInterface Program Code:

```
using System;

public class Program
{
    public static void Main()
    {
        Console.Write("Enter the letter found in the paper: ");
        char letter = Convert.ToChar(Console.ReadLine());
        switch(letter)
        {
            case 'T' :
                IOpenable treasure = new TreasureBox();
                Console.WriteLine(treasure.OpenSesame());
                break;
            case 'P' :
                IOpenable parachute = new Parachute();
                Console.WriteLine(parachute.OpenSesame());
                break;
        }
    }
}

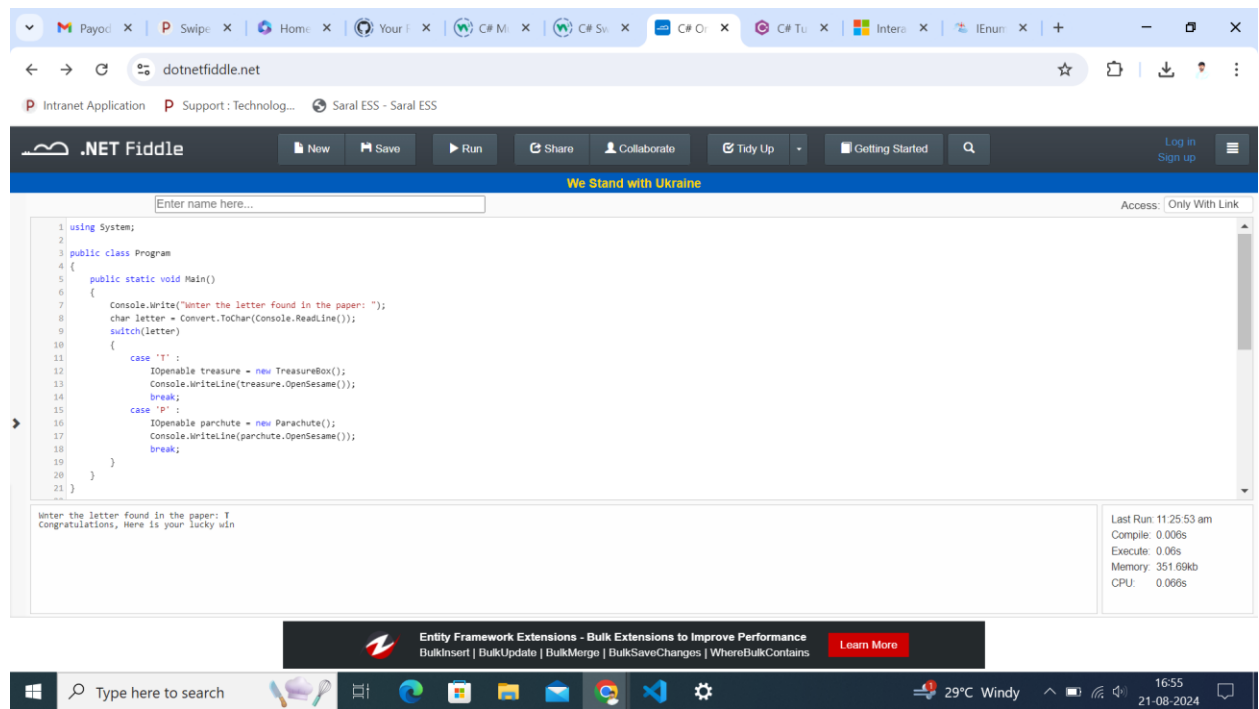
interface IOpenable
{
    string OpenSesame();
}

class TreasureBox : IOpenable
{
    public string OpenSesame()
    {
        return "Congratulations, Here is your lucky win";
    }
}
```

```
}
```

```
class Parachute : IOpenable
{
    public string OpenSesame()
    {
        return "Have a thrilling experience flying in air";
    }
}
```

## Output:



## 2. FlightStatus Program Code:

```
using System;
using System.Collections.Generic;

public class Program
{
    static Dictionary<string, DateTime> flightSchedules = new Dictionary<string,
    DateTime>()
```

```

{
    { "Ar456", new DateTime(DateTime.Now.Year, DateTime.Now.Month,
DateTime.Now.Day, 18, 0, 0) },
    { "Hq457", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 23, 22,
30, 0) },
    { "Tg458", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 22, 20, 15,
0) },
    { "Rs459", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 28, 9, 15,
0) },
    { "Dy460", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 24, 8, 23,
0) },
    { "Zi454", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 23, 5, 40,
0) },
    { "Bm487", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 22, 9, 19,
0) },
};

```

```

public static string FlightStatus(string flightNo)
{
    if (flightSchedules.TryGetValue(flightNo, out DateTime departureTime))
    {
        TimeSpan timeLeft = departureTime - DateTime.Now;
        if (timeLeft.TotalSeconds > 0)
        {
            int days = timeLeft.Days;
            TimeSpan time = timeLeft - TimeSpan.FromDays(days);
            //return $"Time To Flight {days} days {time.ToString(@"hh\:mm\:ss\.ffffff")}";
            //Output
            //Enter the Flight Number: Rs459
            //Time To Flight 6 days 19:58:48.952218
            return $"Time To Flight {timeLeft}";
            //Output
            //Enter the Flight Number: Rs459
            //Time To Flight 6.19:55:58.7112418
        }
        else
        {
            return "Flight Already Left";
        }
    }
}

```

```

    }
}
else
{
    return "Flight Not Found";
}
}

public static void Main()
{
    Console.WriteLine("Enter the Flight Number: ");
    string flightNo = Console.ReadLine();
    Console.WriteLine(FlightStatus(flightNo));
}
}

```

## Output:

The screenshot shows the .NET Fiddle web application interface. The browser address bar displays 'dotnetfiddle.net'. The application has a dark theme and a top navigation bar with buttons for 'Now', 'Save', 'Run', 'Share', 'Collaborate', 'Tidy Up', and 'Getting Started'. A banner for 'We Stand with Ukraine' is visible. On the left, there are settings for 'Language' (C#), 'Project Type' (Console), 'Compiler' (NET 8), and 'NuGet Packages'. The main editor area contains C# code for a flight status program. The output console shows the result of running the program: 'Enter the flight Number: R6459' and 'Time To Flight 6.19:55:58.7112418'. A performance summary on the right indicates the last run was at 1:19:01 pm, with compile time of 0.007s, execute time of 0.07s, memory usage of 1.22Mb, and CPU usage of 0.077s. At the bottom, there is a Windows taskbar with various icons and a system tray showing '29°C Partly cloudy' and the date '21-08-2024'.

```

1 using System;
2 using System.Collections.Generic;
3
4 public class Program
5 {
6     static Dictionary<string, DateTime> flightSchedules = new Dictionary<string, DateTime>()
7     {
8         { "Ar456", new DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day, 18, 0, 0) },
9         { "Hq457", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 23, 22, 30, 0) },
10        { "Tg458", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 22, 20, 15, 0) },
11        { "R6459", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 28, 0, 15, 0) },
12        { "Dy460", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 24, 0, 23, 0) },
13        { "Zl464", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 23, 5, 40, 0) },
14        { "Bm487", new DateTime(DateTime.Now.Year, DateTime.Now.Month, 22, 9, 19, 0) },
15    };
16
17    public static string FlightStatus(string flightNo)
18    {
19
20    }
21 }

```

Enter the flight Number: R6459  
Time To Flight 6.19:55:58.7112418

Last Run: 1:19:01 pm  
Compile: 0.007s  
Execute: 0.07s  
Memory: 1.22Mb  
CPU: 0.077s

## 3. ProductDetails Program Code:

```

using System;
using System.Collections.Generic;

```

```

public class Program
{
    public static void Main()
    {
        List<Product> list = new List<Product>();
        list.Add(new Product("Apple", "Prod1234", DateTime.Now, 200));
        list.Add(new Product("Hair Trimmer", "Prod1276", DateTime.Now, 1200));
        list.Add(new Product("Steel Box", "Prod1209", DateTime.Now, 400));
        list.Add(new Product("Rope", "Prod1213", DateTime.Now, 99));
        list.Add(new Product("Chair", "Prod1342", DateTime.Now, 309));

        Console.WriteLine(String.Format("{0,-15}{1,-15}{2,-25}{3,-15}", "Product Name",
            "Serial Number", "Purchase Date", "Purchase Cost"));
        list.ForEach( a => Console.WriteLine(a));

    }
}

class Product
{
    string _productName;
    string _serialNumber;
    DateTime _purchaseDate;
    double _cost;

    public Product(string productName, string serialNumber, DateTime purchaseDate,
        double cost)
    {
        _productName = productName;
        _serialNumber = serialNumber;
        _purchaseDate = purchaseDate;
        _cost = cost;
    }

    public override string ToString()
    {

```

```

return String.Format("{0,-15}{1,-15}{2,-25}{3,-15}", _productName, _serialNumber,
_purchaseDate.ToString(@"yyyy:mm:dd"), _cost);
}

}

```

## Output:

The screenshot shows the .NET Fiddle web application interface. The browser address bar displays `dotnetfiddle.net`. The application has a dark theme and a top navigation bar with buttons for New, Save, Run, Share, Collaborate, Tidy Up, and Getting Started. A banner for "We Stand with Ukraine" is visible. The main area is divided into three sections: Options, Code, and Output.

**Options:**

- Language: C#
- Project Type: Console
- Compiler: .NET 8
- NuGet Packages: (empty)
- Auto Run: ☒ Yes ☐ No

**Code:**

```

1 using System;
2 using System.Collections.Generic;
3
4 public class Program
5 {
6     public static void Main()
7     {
8         List<Product> list = new List<Product>();
9         list.Add(new Product("Apple", "Prod1234", DateTime.Now, 200));
10        list.Add(new Product("Hair Trimmer", "Prod1276", DateTime.Now, 1200));
11        list.Add(new Product("Steel Box", "Prod1289", DateTime.Now, 400));
12        list.Add(new Product("Rope", "Prod1213", DateTime.Now, 99));
13        list.Add(new Product("Chair", "Prod1342", DateTime.Now, 300));
14
15        Console.WriteLine(String.Format("{0,-15}{1,-15}{2,-25}{3,-15}", "Product Name", "Serial Number", "Purchase Date", "Purchase Cost"));
16        list.ForEach(a => Console.WriteLine(a));
17    }
18 }

```

**Output:**

Product Name	Serial Number	Purchase Date	Purchase Cost
Apple	Prod1234	2024-08-21	200
Hair Trimmer	Prod1276	2024-08-21	1200
Steel Box	Prod1289	2024-08-21	400
Rope	Prod1213	2024-08-21	99
Chair	Prod1342	2024-08-21	300

Performance metrics: Last Run 1:28:00 pm, Compile: 0.007s, Execute: 0.08s, Memory: 1.47Mb, CPU: 0.087s.

At the bottom, there is a banner for "Entity Framework Extensions - Bulk Extensions to Improve Performance" with links for BulkInsert, BulkUpdate, BulkMerge, BulkSaveChanges, and WhereBulkContains.