

## NB Healthcare Technologies Pvt Ltd

### Day 18 Morning Assignment (16 – Feb- 2022)

By

Vamsi Krishna Mandapati

#### 1. What is the use of XML

##### Uses Of XML:

- XML – eXtensible Markup Language.
- XML is used for universal data transfer mechanism to send data across different platforms.
- XML is one of the most widely-used formats **for sharing structured information today**: between programs, between people, between computers and people, both locally and across networks.
- XML plays an important role in many different IT systems.
- XML is often used for distributing data over the Internet.
- It is important (for all types of software developers!) to have a good understanding of XML.

XML has a variety of uses for Web, e-business, and portable applications.

The following are some of the many applications for which XML is useful:

- **Web publishing:** XML allows you to create interactive pages, allows the customer to customize those pages, and makes creating e-commerce applications more intuitive. With XML, you store the data once and then render that content for different viewers or devices based on style sheet processing using an Extensible Style Language (XSL)/XSL Transformation (XSLT) processor.
- **Web searching and automating Web tasks:** XML defines the type of information contained in a document, making it easier to return useful results when searching the Web:

For example, using HTML to search for books authored by Tom Brown is likely to return instances of the term 'brown' outside of the context of author. Using XML restricts the search to the correct context (for example, the information

contained in the <author> tag) and returns only the information that you want. By using XML, Web agents and robots (programs that automate Web searches or other tasks) are more efficient and produce more useful results.

- **General applications:** XML provides a standard method to access information, making it easier for applications and devices of all kinds to use, store, transmit, and display data.
  - **e-business applications:** XML implementations make electronic data interchange (EDI) more accessible for information interchange, business-to-business transactions, and business-to-consumer transactions.
  - **Metadata applications:** XML makes it easier to express metadata in a portable, reusable format.
  - **Pervasive computing:** XML provides portable and structured information types for display on pervasive (wireless) computing devices such as personal digital assistants (PDAs), cellular phones, and others. For example, WML (Wireless Markup Language) and Voice are currently evolving standards for describing visual and speech-driven wireless device interfaces.
- It simplifies data sharing
  - It simplifies data transport
  - It simplifies platform changes
  - It simplifies data availability

Many computer systems contain data in incompatible formats. Exchanging data between incompatible systems (or upgraded systems) is a time-consuming task for web developers. Large amounts of data must be converted, and incompatible data is often lost.

XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data.

XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

With XML, data can be available to all kinds of "reading machines" like people, computers, voice machines, news feeds, etc.

## 2. Write the points discussed about xml in the class

### XML:

- XML :- eXtensible Markup Language.
- XML Tags are user defined.
- XML is case sensitive
- XML is having only one Root tag

#### Syntax:

<Employees>

-----

-----

</Employees>

- If We write two root tags in xml file it will error.( the syntax rules of XML are strict: XML tools will not process files that contain errors, but instead will give you error messages so that you fix them)
- There are two types of XML Files
  - 1)Tag Based XML :
    - Tag Based XML will take more memory/size. Those who don't think about size they still go for tag based XML

#### Example:

<Products>

<Laptop>

<brand>HP</brand>

<price>50000</price>

<ram>8GB</ram>

</Laptop>

</products>

#### 2) Attribute Based XML

- Tag based XML consume more size, that's why attribute based XML was Introduced. Attribute based XML consume less size than tag-based XML.

#### Example:

<Products>

<Laptop brand = "HP" price = "50000" ram = "8GB" </Laptop>

</products>

## 3. Create a simple xml to illustrate:

- a. Tag based xml with 10 products
- b. Attribute based xml

### A. Tag Based XML

```
<products>
  <laptop>
    <brand>HP</brand>
    <price>50000</price>
    <ram>8GB</ram>
  </laptop>

  <mobile>
    <brand>Vivo</brand>
    <price>20000</price>
    <ram>4GB</ram>
  </mobile>

  <shirt>
    <brand>John Players</brand>
    <price>5000</price>
    <color>white</color>
  </shirt>

  <tv>
    <brand>Samsung</brand>
    <price>25000</price>
    <color>black</color>
  </tv>

  <bike>
    <brand>Royal Enfield</brand>
    <price>200000</price>
    <color>Grey</color>
  </bike>

  <car>
    <brand>BMW</brand>
    <price>2000000</price>
    <color>Blue</color>
  </car>

  <bicycle>
    <brand>Hero</brand>
    <price>2000</price>
    <color>Black</color>
  </bicycle>

  <mouse>
    <brand>Dell</brand>
    <price>150</price>
    <color>Black</color>
```

```
</mouse>
```

```
<keyboard>
```

```
<brand>HP</brand>
```

```
<price>200</price>
```

```
<color>Black</color>
```

```
</keyboard>
```

```
<watch>
```

```
<brand>Apple</brand>
```

```
<price>1900</price>
```

```
<color>Black</color>
```

```
</watch>
```

```
</products>
```

#### B. Attribute Based XML

```
<products>
```

```
<Laptop brand = "HP" price = "50000" ram = "8GB" ></Laptop>
```

```
<Mobile brand = "Vivo" price = "20000" ram = "4GB" ></Mobile>
```

```
<shirt brand = "John Players" price = "5000" color = "White"></shirt>
```

```
<tv brand = "Samsung" price = "25000" color = "black" ></tv>
```

```
<bike brand = "Royal Enfield" price = "200000" color = "Grey" ></bike>
```

```
<car brand = "BMW" price = "2000000" color = "Blue" ></car>
```

```
<bicycle brand = "Hero" price = "2000" color = "Black"></bicycle>
```

```
<mouse brand = "HP" price = "150" color = "Black" ></mouse>
```

```
<keyboard brand = "HP" price = "200" color = "Black"></keyboard>
```

```
<watch brand = "Apple" price = "1900" color = "Black"></watch>
```

```
</products>
```

#### 4. Convert the above xml to JSON and display the JSON data

```
{
  "products": {
    "laptop": {
      "brand": "HP",
      "price": "50000",
      "ram": "8GB"
    },
    "mobile": {
      "brand": "Vivo",
      "price": "20000",
      "ram": "4GB"
    },
    "shirt": {
      "brand": "John Players",
      "price": "5000",
      "color": "white"
    },
    "tv": {
      "brand": "Samsung",
```

```

        "price": "25000",
        "color": "black"
    },
    "bike": {
        "brand": "Royal Enfield",
        "price": "200000",
        "color": "Grey"
    },
    "car": {
        "brand": "BMW",
        "price": "2000000",
        "color": "Blue"
    },
    "bicycle": {
        "brand": "Hero",
        "price": "2000",
        "color": "Black"
    },
    "mouse": {
        "brand": "Dell",
        "price": "150",
        "color": "Black"
    },
    "keyboard": {
        "brand": "HP",
        "price": "200",
        "color": "Black"
    },
    "watch": {
        "brand": "Apple",
        "price": "1900",
        "color": "Black"
    }
}

```

##### 5. Research and write the benefits of JSON over XML ( 2 or 3 points )

###### **Benefits of JSON over XML:**

- JSON – JavaScript Object Notation.
- JSON Stores data as Key – Value Pairs.
- Both JSON and XML can be used to receive data from a web server.
- XML has to be parsed with an XML parser. XML is much more difficult to parse than JSON. JSON is much easier to parse.
- JSON can be parsed by a standard JavaScript function.
- JSON is parsed into a ready-to-use JavaScript object.

- In most scenarios, JSON is undoubtedly easier to read in its expanded form than XML.
- JSON can have a substantially lower character count reducing the overhead in data transfers.
- **JSON is faster because it is designed specifically for data interchange.** XML is slower, because it is designed for a lot more than just data interchange.
- JSON is simple text. ...
- JSON is compact. ...
- JSON is easy to learn, easy to read, and easy to understand.
- We use JSON because **it's extremely lightweight to send back and forth in HTTP requests and responses due to the small file size.** It's easy to read compared to something like XML since it's much cleaner and there's not as many opening and closing tags to worry about.

6. For the below requirement, create a layered architecture project with separate class library for Business logic.

create console application

create windows(or desktop) application

Business Requirement:

FIND FACTORIAL OF A NUMBER:

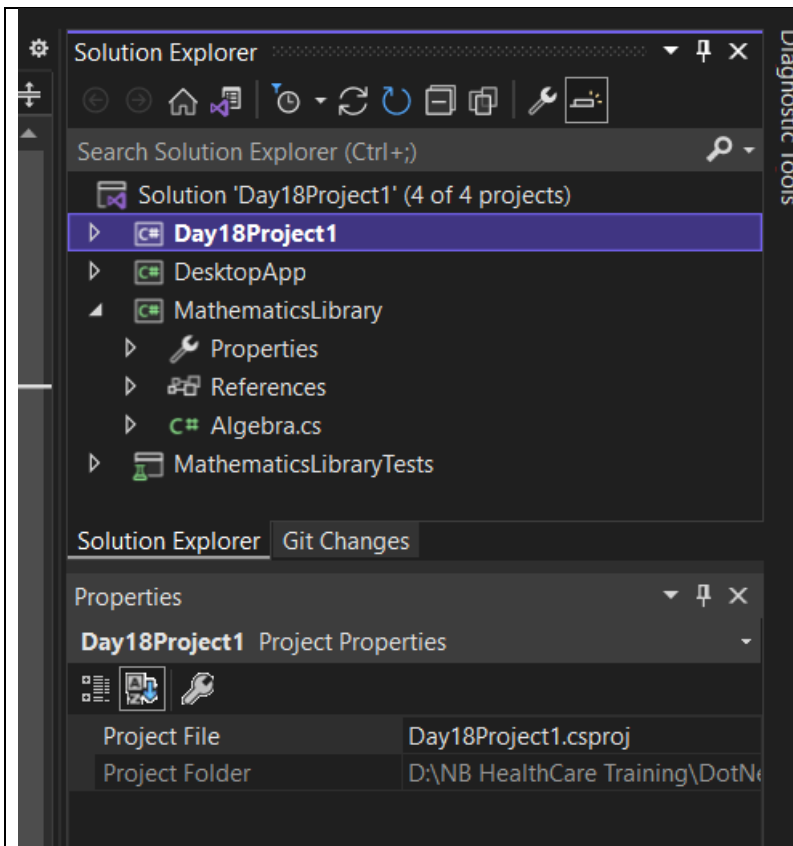
0 = 1

positive number (upto 7) = factorial answer

> 7 = -999 (as answer)

< 0 = -9999 (as answer)

put the screen shots of the output and  
project (solution explorer) screen shot



### Mathematics Library/Class Library For Business Logic

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

namespace MathematicsLibrary
{
    public class Algebra
    {
        public static int Factorial(int n)
        {
            int fact = 1;
            if (n == 0)
                return 1;
            else if (n > 7)
                return -999;
            else if (n < 0)
                return -9999;
            else
            {
                for(int i = 1; i <= n; i++)
                    fact = fact * i;
                return fact;
            }
        }
    }
}
```



```
}
```

## Console App:

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MathematicsLibrary;

namespace Day18Project1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int n;

            Console.WriteLine("Enter any Number:");
            n = Convert.ToInt32(Console.ReadLine());

            Console.WriteLine(MathematicsLibrary.Factorial(n));

            Console.ReadLine();
        }
    }
}
```

Output:

D:\NB HealthCare Training\DotNet Projects\Day 18 Morning Assignment\Day18Project1\Day18Project1\bin\Debug\Day18Project1.exe

```
Enter any Number:
4
24
```

D:\NB HealthCare Training\DotNet Projects\Day 18 Morning Assignment\Day18Project1\Day18Project1\bin\Debug\Day18Project1.exe

```
Enter any Number:
0
1
```

D:\NB HealthCare Training\DotNet Projects\Day 18 Morning Assignment\Day18Project1\Day18Project1\bin\Debug\Day18Project1.exe

Enter any Number:

50

-999

■

D:\NB HealthCare Training\DotNet Projects\Day 18 Morning Assignment\Day18Project1\Day18Project1\bin\Debug\Day18Project1.exe

Enter any Number:

-3

-9999

■

Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'Day18Project1' (4 of 4 projects)

- Day18Project1
  - Properties
  - References
  - App.config
  - C# Program.cs
  - DesktopApp
  - MathematicsLibrary
  - MathematicsLibraryTests

Solution Explorer | Git Changes

Properties

Day18Project1 Project Properties

Project File	Day18Project1.csproj
Project Folder	D:\NB HealthCare Training\DotNet Projects\Day 18 Morning Assignment\Day18Project1\Day18Project1

Windows App:

Code:

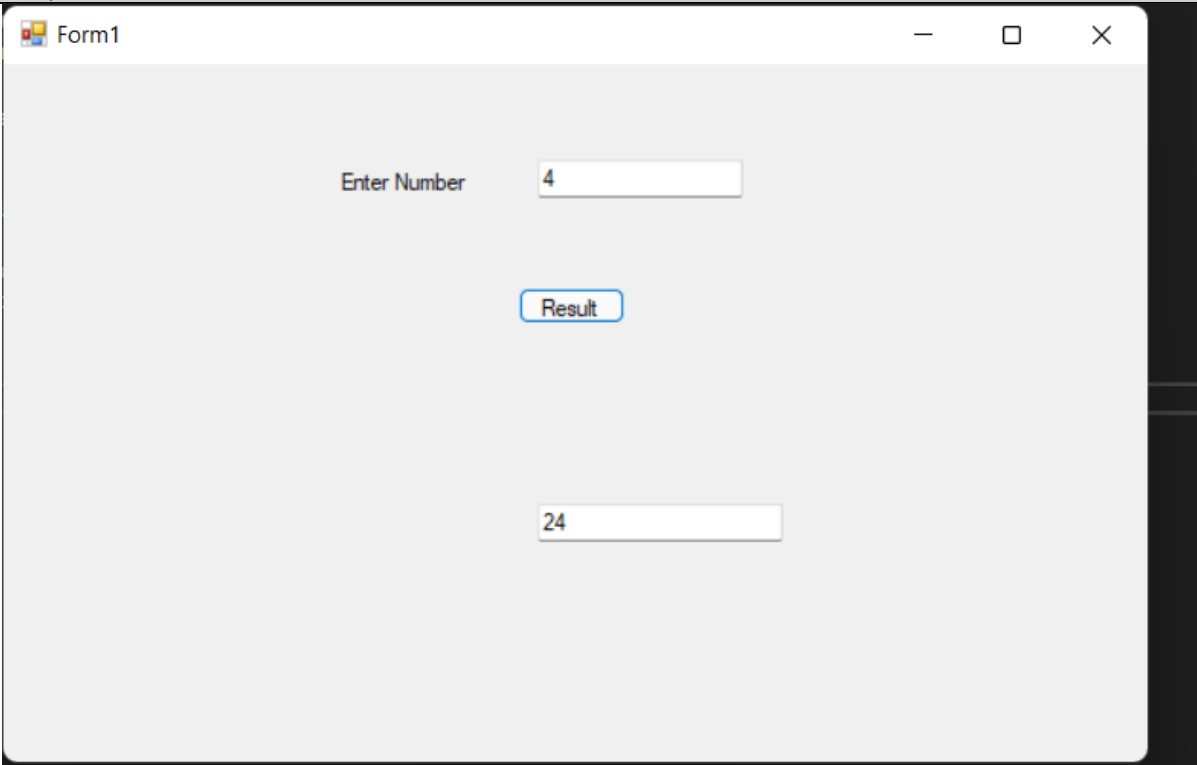
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MathematicsLibrary;

namespace DesktopApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int n = Convert.ToInt32(textBox1.Text);
            int result = Algebra.Factorial(n);
            textBox2.Text = result.ToString();
        }
    }
}
```

Output:



The screenshot shows a Windows application window titled "Form1". Inside the window, there is a label "Enter Number" next to a text input field that contains the value "4". Below the input field is a button labeled "Result". At the bottom of the window, there is another text input field that displays the value "24".

Form1

Enter Number

Form1

Enter Number

Form1

Enter Number

Solution Explorer

Search Solution Explorer (Ctrl+;)

- Day18Project1
  - DesktopApp
    - Properties
    - References
    - App.config
    - Form1.cs
    - Program.cs
  - MathematicsLibrary
  - MathematicsLibraryTests

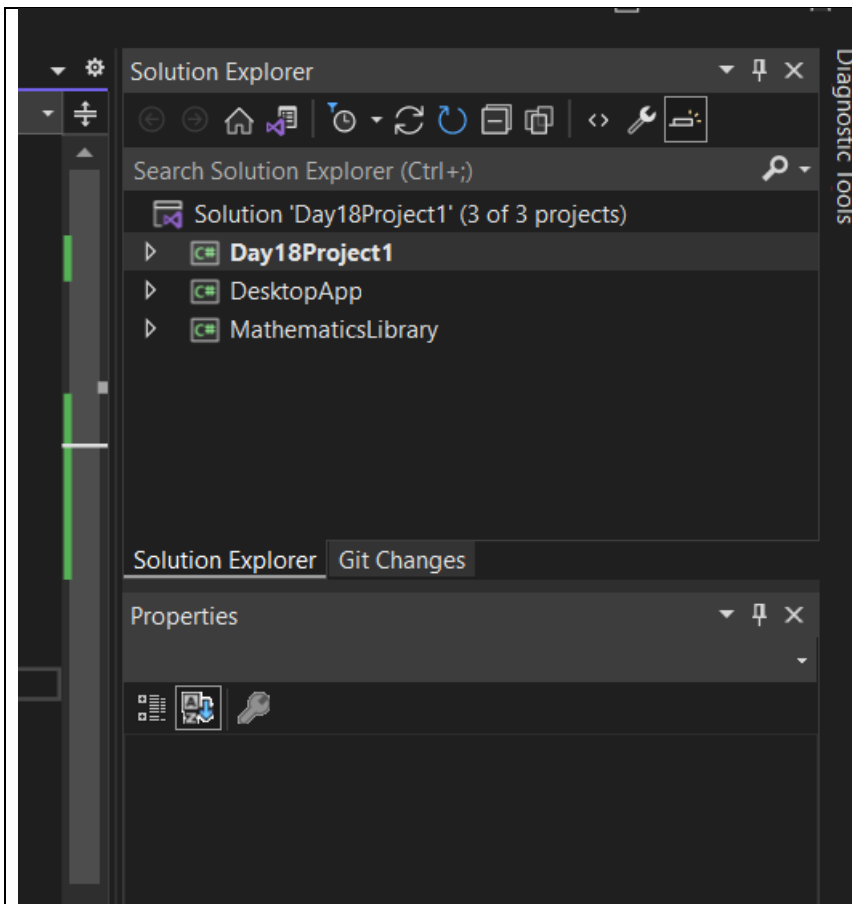
Solution Explorer Git Changes

Properties

Day18Project1 Project Properties

Project File	Day18Project1.csproj
Project Folder	D:\NB HealthCare Training\DotNe

Project Solution Explorer



7. For the above method, Implement TDD  
and write 4 test cases and put the code in word document.  
put the screen shot of all test cases failing.

make the test cases pass.

put the screen shot

Code:

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

namespace MathematicsLibrary
{
    public class Algebra
    {
        public static int Factorial(int n)
        {
            int fact = 1;
            if (n == 0)
                return 1;
            else if (n > 7)
                return -999;
            else if (n < 0)
```

```

        return -9999;
    else
    {
        for (int i = 1; i <= n; i++)
            fact = fact * i;
        return fact;
    }
}
}

```

#### Test Cases:

```

using Microsoft.VisualStudio.TestTools.UnitTesting;
using MathematicsLibrary;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MathematicsLibrary.Tests
{
    [TestClass()]
    public class AlgebraTests
    {
        [TestMethod()]
        public void FactorialTest_Zero_Input()
        {
            //Arrange
            int n = 0;
            int expected = 1;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]
        public void FactorialTest_one_to_seven_Input()
        {
            //Arrange
            int n = 5;
            int expected = 120;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]
        public void FactorialTest_Negative_Input()
        {
            //Arrange
            int n = -1;
            int expected = -9999;

            //Act

```

```

        int actual = Algebra.Factorial(n);

        //Assert
        Assert.AreEqual(expected, actual);
    }

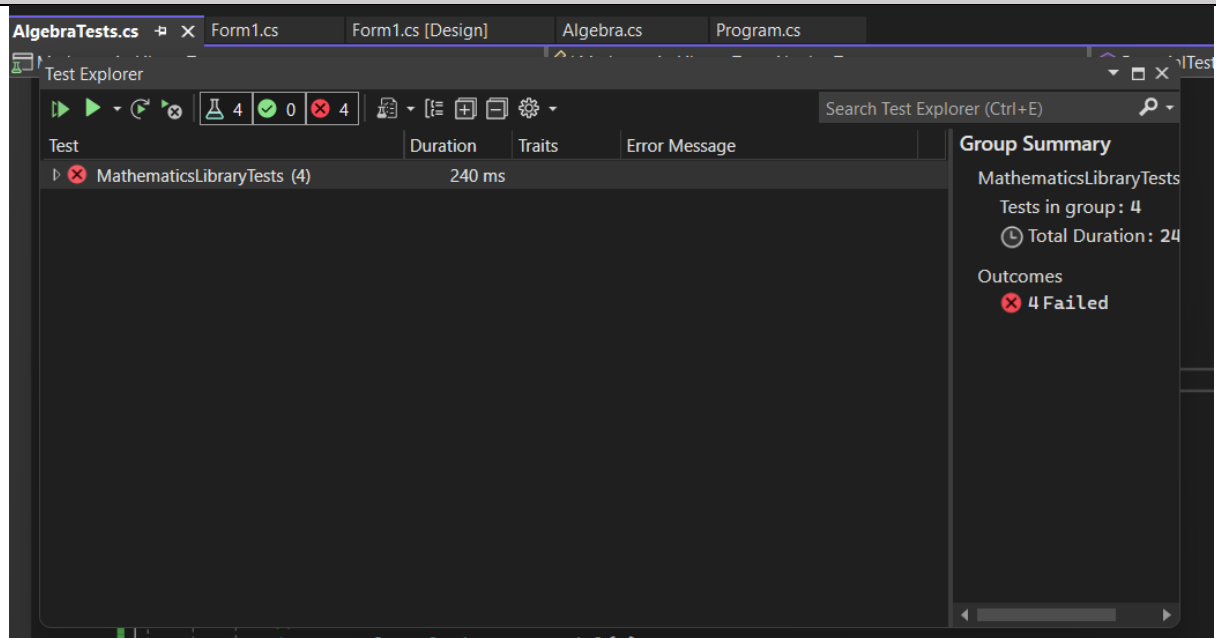
    [TestMethod()]
    public void FactorialTest_greater_than_seven_Input()
    {
        //Arrange
        int n = 8;
        int expected = -999;

        //Act
        int actual = Algebra.Factorial(n);

        //Assert
        Assert.AreEqual(expected, actual);
    }
}

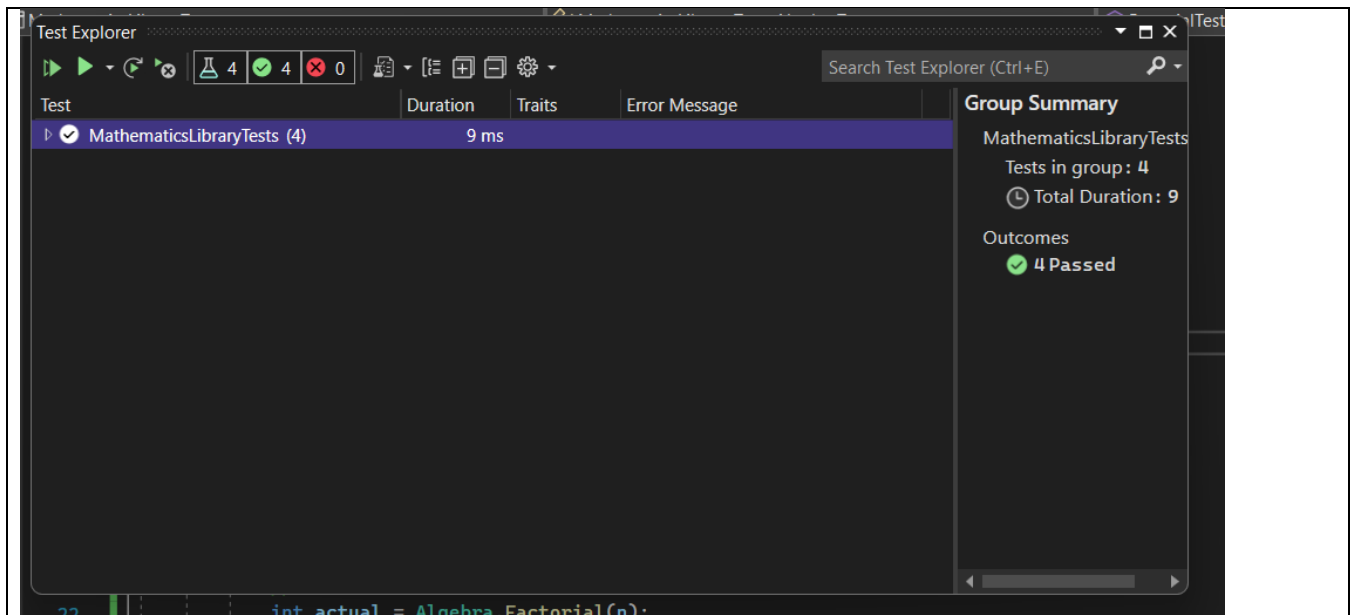
```

4 Test Cases Failed:



4 Test Cases are Passed:





8. Add one more method to check if the number is palindrome or not in the above Algebra class and write test case for the same.

Code:

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

namespace MathematicsLibrary
{
    public class Algebra
    {
        public static int Factorial(int n)
        {
            int fact = 1;
            if (n == 0)
                return 1;
            else if (n > 7)
                return -999;
            else if (n < 0)
                return -9999;
            else
            {
                for (int i = 1; i <= n; i++)
                    fact = fact * i;
                return fact;
            }
        }

        public static bool IsPalindrome(int n)
        {
            int m, rem, rev = 0;

            Console.WriteLine("Enter any Number");
            n = Convert.ToInt32(Console.ReadLine());
```

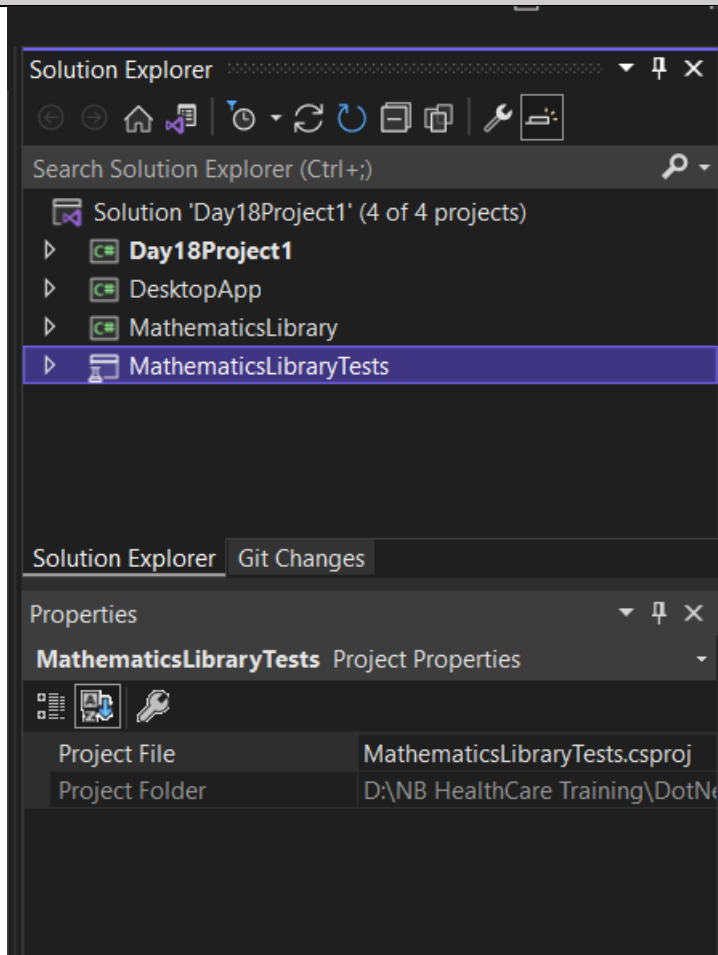
```

        m = n;
        while (m > 0)
        {
            rem = m % 10;
            m = m / 10;
            rev = rev * 10 + rem;

        }
        if (n == rev)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

```

## Test Cases



```

using Microsoft.VisualStudio.TestTools.UnitTesting;
using MathematicsLibrary;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace MathematicsLibrary.Tests
{
    [TestClass()]
    public class AlgebraTests
    {
        [TestMethod()]
        public void FactorialTest_Zero_Input()
        {
            //Arrange
            int n = 0;
            int expected = 1;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]
        public void FactorialTest_one_to_seven_Input()
        {
            //Arrange
            int n = 5;
            int expected = 120;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]
        public void FactorialTest_Negative_Input()
        {
            //Arrange
            int n = -1;
            int expected = -9999;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]
        public void FactorialTest_greater_than_seven_Input()
        {
            //Arrange
            int n = 8;
            int expected = -999;

            //Act
            int actual = Algebra.Factorial(n);

            //Assert
            Assert.AreEqual(expected, actual);
        }

        [TestMethod()]

```

```
public void PalindromeTest_Input()
{
    //Arrange
    int n = 95;
    bool expected = true;

    //Act
    int actual = Algebra.IsPalindrome(n);

    //Assert
    Assert.AreEqual(expected, actual);
}
}
```