Day 18 Assignment
by
Vinay Kudali
16-02-2022

NB Healthcare Technologies

1. What is the use of XML

Xml is used for universal data transfer mechanism to send data across different platforms.

2. Write the points discussed about xml in the class

1. Xml will have user defined tags.
2. Xml is case sensitive.
3. Xml have only one root tag.
4. Xml is used for universal data transfer mechanism to send data across different platforms.
5. In Xml, we have two types: 1) Tag Based xml, 2) Attribute based xml
6. Tag based xml will take a bit more memory size over attribute-based xml.
7. Example for Tag based xml:

```
<Products>
  <product>
    <code> 123</code>
    <name>Battery</name>
    <Brand>Nippon</Brand>
  </product>
</Products>
```

8. Example for Attribute based xml:

```
<Employees>
   <Employee>Id = “321” Name= “Ram Charan” Salary = “2300” />
</Employees>
```

3. Create a simple xml to illustrate:

a. Tag based xml with 10 products

```xml
<products>
  <product1>
    <ProductId>1234</ProductId>
    <ProductName>Notebook</ProductName>
    <ProductBrand>Classmate</ProductBrand>
  </product1>
  <product2>
    <ProductId>2341</ProductId>
    <ProductName>Pen</ProductName>
    <ProductBrand>Cello</ProductBrand>
  </product2>
  <product3>
    <ProductId>9876</ProductId>
    <ProductName>Laptop</ProductName>
    <ProductBrand>Msi</ProductBrand>
  </product3>
  <product4>
    <ProductId>1267</ProductId>
    <ProductName>T.V</ProductName>
    <ProductBrand>Onida</ProductBrand>
  </product4>
  <product5>
    <ProductId>8465</ProductId>
    <ProductName>Choclate</ProductName>
    <ProductBrand>DiaryMilk</ProductBrand>
  </product5>
  <product6>
    <ProductId>9867</ProductId>
    <ProductName>Shirt</ProductName>
    <ProductBrand>Netplay</ProductBrand>
  </product6>
  <product7>
    <ProductId>5678</ProductId>
    <ProductName>Watch</ProductName>
    <ProductBrand>Sonata</ProductBrand>
  </product7>
  <product8>
    <ProductId>4356</ProductId>
    <ProductName>Mobile</ProductName>
    <ProductBrand>Xiaomi</ProductBrand>
  </product8>
  <product9>
    <ProductId>4942</ProductId>
    <ProductName>Mouse</ProductName>
    <ProductBrand>Logitech</ProductBrand>
  </product9>
  <product10>
    <ProductId>4120</ProductId>
    <ProductName>Spects</ProductName>
    <ProductBrand>Lenskart</ProductBrand>
  </product10>
</products>
```

b. Attribute based xml

```xml
▼<students>
   <student1 ID="19267" name="hardik" branch="Electronics and comunications"/>
   <student2 ID="13628" name="jadeja" branch="Mechanical"/>
   <student3 ID="90893" name="prudhvisha" branch="Electrical"/>
   <student4 ID="73832" name="harabajan" branch="Information Technology"/>
   <student5 ID="67236" name="vinay" branch="computer science"/>
   <student6 ID="77822" name="kuldeep" branch="Civil"/>
   <student7 ID="76873" name="virat" branch="Automobile"/>
   <student8 ID="76712" name="dhoni" branch="Petrolium"/>
   <student9 ID="76912" name="sachin" branch="Mining"/>
   <student10 ID="76561" name="rohit" branch="Chemical"/>
 </students>
```


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


```json
{
 "students": {
  "student1": {
   "-ID": "19267",
   "-name": "hardik",
   "-branch": "Electronics and comunications",
   "-self-closing": "true"
  },
  "student2": {
   "-ID": "13628",
   "-name": "jadeja",
   "-branch": "Mechanical",
   "-self-closing": "true"
  },
  "student3": {
   "-ID": "90893",
   "-name": "prudhvisha",
   "-branch": "Electrical",
   "-self-closing": "true"
  },
  "student4": {
   "-ID": "73832",
   "-name": "harabajan",
   "-branch": "Information Technology",
   "-self-closing": "true"
  },
  "student5": {
   "-ID": "67236",
```

```
      "-name": "vinay",
      "-branch": "computer science",
      "-self-closing": "true"
    },
    "student6": {
      "-ID": "77822",
      "-name": "kuldeep",
      "-branch": "Civil",
      "-self-closing": "true"
    },
    "student7": {
      "-ID": "76873",
      "-name": "virat",
      "-branch": "Automobile",
      "-self-closing": "true"
    },
    "student8": {
      "-ID": "76712",
      "-name": "dhoni",
      "-branch": "Petrolium",
      "-self-closing": "true"
    },
    "student9": {
      "-ID": "76912",
      "-name": "sachin",
      "-branch": "Mining",
      "-self-closing": "true"
    },
    "student10": {
      "-ID": "76561",
      "-name": "rohit",
      "-branch": "Chemical",
      "-self-closing": "true"
    }
  },
  "#omit-xml-declaration": "yes"
}
```

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

1. JavaScript Object Notation will take less memory size over xml.
2. JSON has no Tags.
3. JSON is Much easier to parse.
4. JSON has a lower character count reducing the overhead in data transfers.

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

Business Requirement: FINDING FACTORIAL OF A NUMBER: 0 = 1 positive number (up to 7) = factorial answer > 7 = -999 (as answer) < 0 = -9999 (as answer) put the screen shots of the output and project (solution explorer) screen shot

Code:

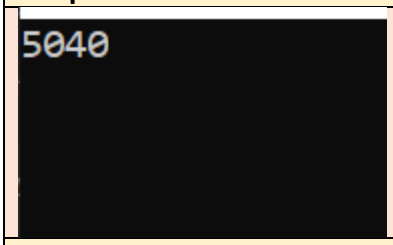
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MathsLibrary;

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

```
            }
            return fact;
        }
    }
  }
}
```

create console application

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

namespace ConsoleApp
{
  internal class Program
  {
    static void Main(string[] args)
    {

      Console.WriteLine(AlgebraClass.Factorial(7));
      Console.ReadLine();
    }
  }
}
```

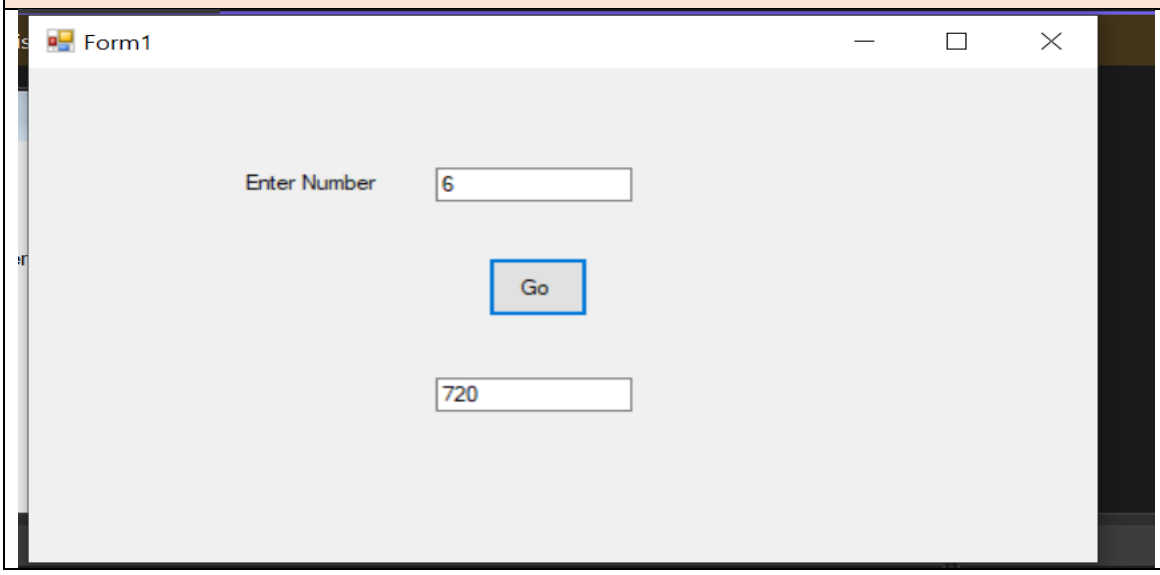
Output:



b. creates windows (or desktop) application

```
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 MathsLibrary;

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

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



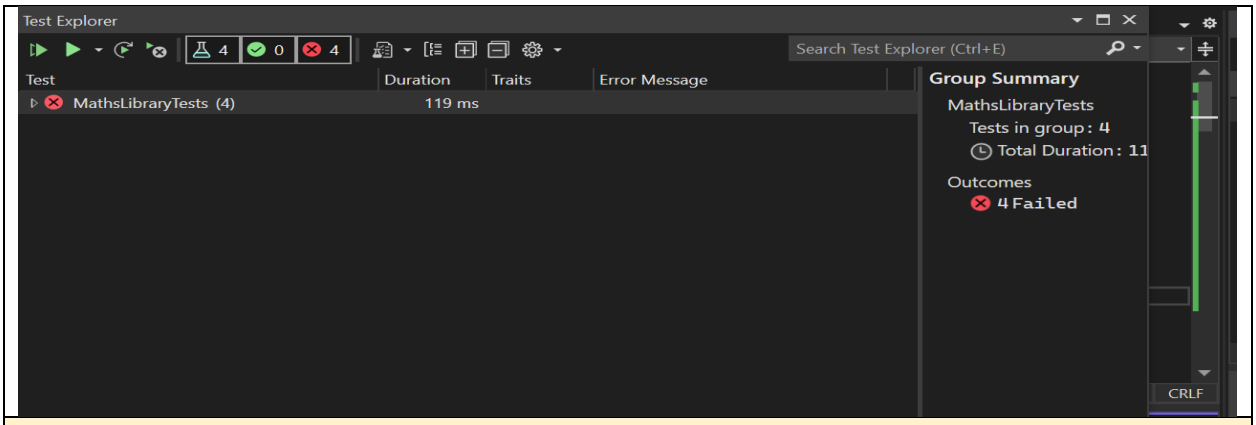
7. For the above method, Implement TDD and write 4 test cases and put the code in word document. put the screen shot of all test cases failing. make the test cases pass. put the screen shot

AlgebraTestClass:

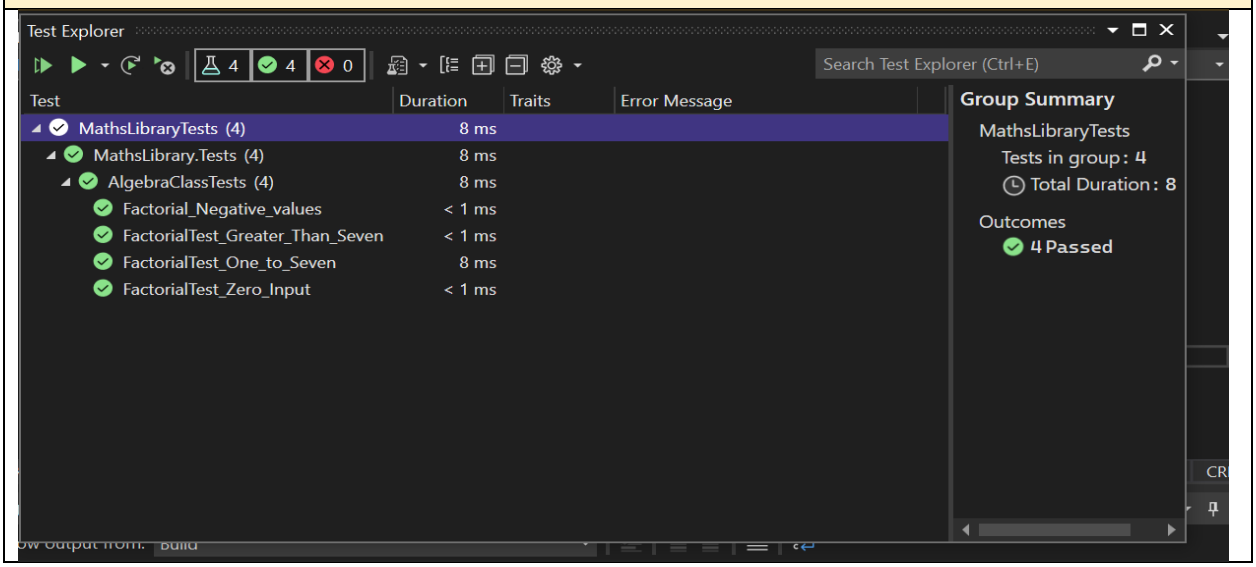
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using MathsLibrary;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MathsLibrary.Tests
{
   [TestClass()]
   public class AlgebraClassTests
   {
      [TestMethod()]
      public void FactorialTest_Zero_Input()
      {
         //Arrange
         int input = 0;
         int expected =1;

         //Act
         int actual =AlgebraClass.Factorial(input);

         //Assert
         Assert.AreEqual(expected, actual);
      }
      [TestMethod()]
      public void FactorialTest_One_to_Seven()
      {
         //Arrange
         int input = 5;
         int expected = 120;

         //Act
         int actual = AlgebraClass.Factorial(input);
```

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

    public void FactorialTest_Greater_Than_Seven()
    {
        //Arrange
        int input = 8;
        int expected =-999;

        //Act
        int actual = AlgebraClass.Factorial(input);

        //Assert
        Assert.AreEqual(expected, actual);
    }
    [TestMethod()]
    public void Factorial_Negative_values()
    {
        //Arrange
        int input = -3;
        int expected = -9999;

        //Act
        int actual = AlgebraClass.Factorial(input);

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

  }
}
```

TestFailed:



TestPass:



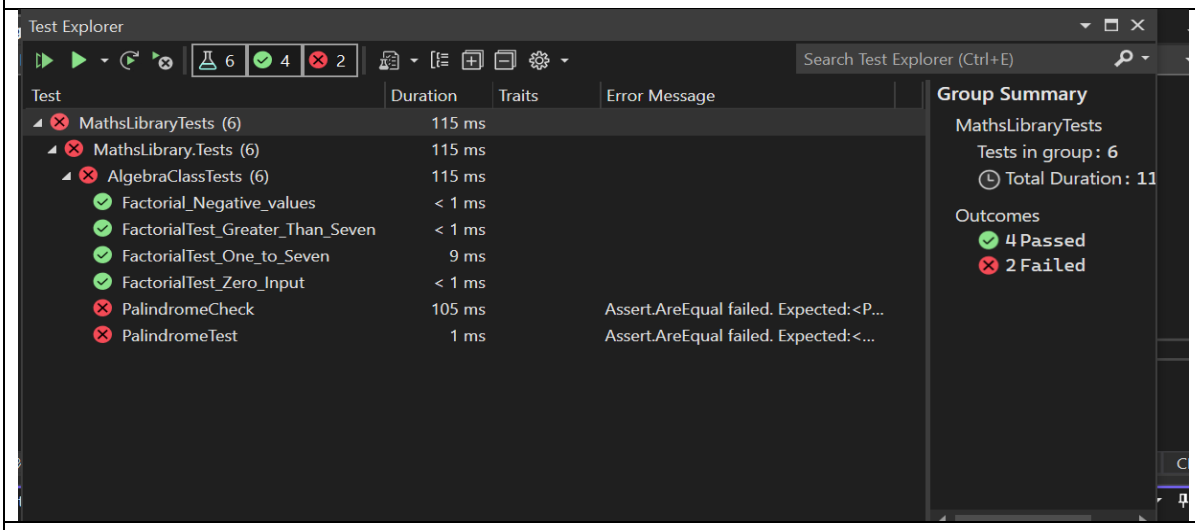
8. Add one more method to check if the number is palindrome or not in the above Algebra class and write test case for the same.

Code:

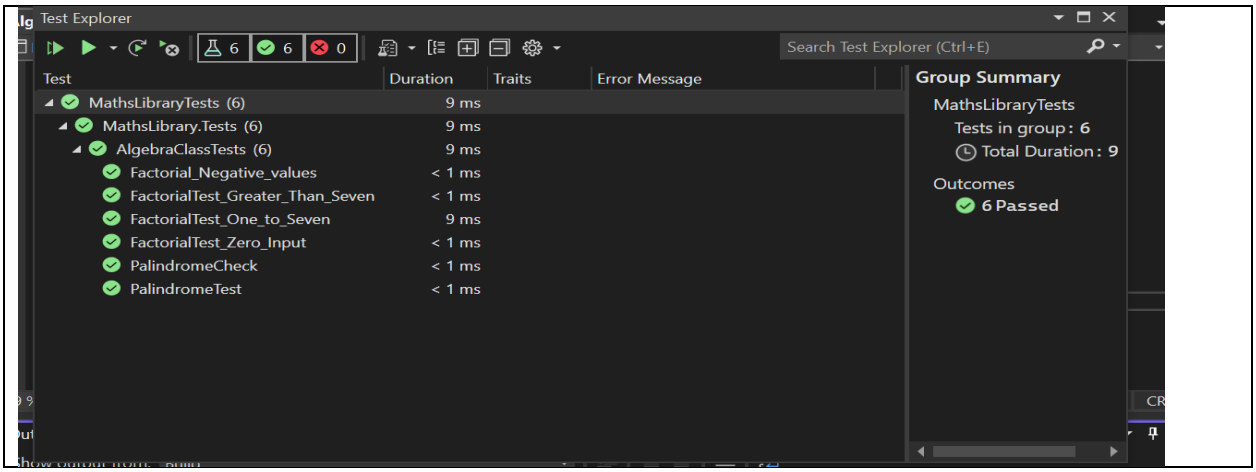
```
[TestMethod()]
public void PalindromeTest()
{
    //Arrange
    int input = 221;
    string expected = "Not Palindrome";

    //Act
    string actual = AlgebraClass.Palindrome(input);
```

```
        //Assert
        Assert.AreEqual(expected, actual);
    }
    [TestMethod()]
    public void PalindromeCheck()
    {
        //Arrange
        int input = 323;
        string expected = "Palindrome";

        //Act
        string actual = AlgebraClass.Palindrome(input);


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

    }
  }
}
```

TestCaseFailed:



TestCasePassed:


Test Explorer
6
6
0
Search Test Explorer (Ctrl+E)
Test
Duration
Traits
Error Message
MathsLibraryTests (6)
9 ms
MathsLibrary.Tests (6)
9 ms
AlgebraClassTests (6)
9 ms
Factorial_Negative_values
< 1 ms
FactorialTest_Greater_Than_Seven
< 1 ms
FactorialTest_One_to_Seven
9 ms
FactorialTest_Zero_Input
< 1 ms
PalindromeCheck
< 1 ms
PalindromeTest
< 1 ms
Group Summary
MathsLibraryTests
Tests in group : 6
Total Duration : 9
Outcomes
6 Passed