Feb 16th Assignment

By

Surya Teja Chandolu

1. What is the use of XML

XML can be used to exchange the information between organizations and systems. XML can be used for offloading and reloading of databases. XML can be used to store and arrange the data, which can customize your data handling needs. XML can easily be merged with style sheets to create almost any desired output.

2. Write the points discussed about xml in the class

- XML means **eXtensible Markup Language**.
- XML is used for universal data transfer mechanism to send data across different platform.
- XML is user defined tags.
- XML has only one root tag.
- XML is case sensitive.

3. Create a simple xml to illustrate:
a. Tag based xml with 10 products
b. Attribute based xml

Code:

TagBased XML:

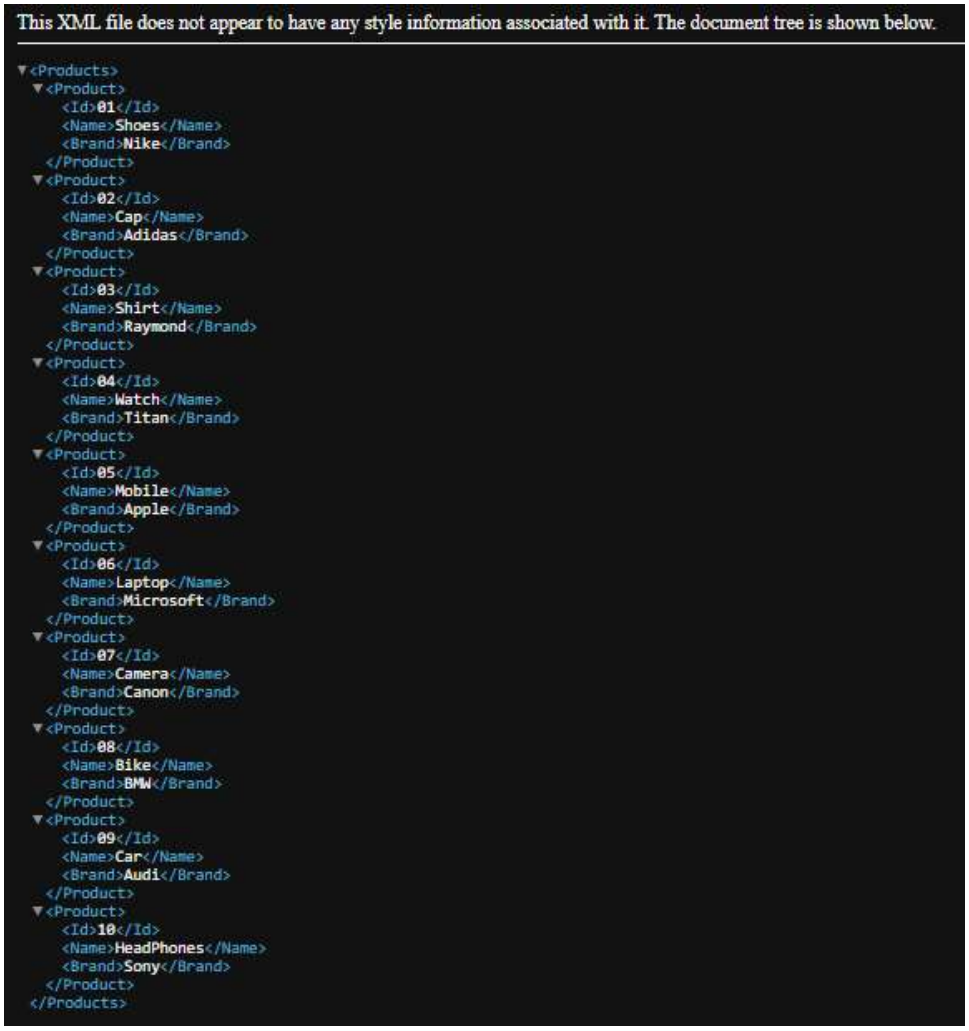
```
<Products>
    <Product>
        <Id>01</Id>
        <Name>Shoes</Name>
        <Brand>Nike</Brand>
    </Product>
    <Product>
        <Id>02</Id>
        <Name>Cap</Name>
        <Brand>Adidas</Brand>
    </Product>
    <Product>
        <Id>03</Id>
        <Name>Shirt</Name>
        <Brand>Raymond</Brand>
    </Product>
    <Product>
        <Id>04</Id>
        <Name>Watch</Name>
        <Brand>Titan</Brand>
    </Product>
    <Product>
        <Id>05</Id>
        <Name>Mobile</Name>
        <Brand>Apple</Brand>
    </Product>
    <Product>
        <Id>06</Id>
        <Name>Laptop</Name>
        <Brand>Microsoft</Brand>
    </Product>
    <Product>
        <Id>07</Id>
        <Name>Camera</Name>
        <Brand>Canon</Brand>
    </Product>
    <Product>
        <Id>08</Id>
        <Name>Bike</Name>
        <Brand>BMW</Brand>
    </Product>
    <Product>
        <Id>09</Id>
        <Name>Car</Name>
        <Brand>Audi</Brand>
    </Product>
    <Product>
        <Id>10</Id>
        <Name>HeadPhones</Name>
        <Brand>Sony</Brand>
    </Product>
</Products>
```

AttributeBased XML:

```
<Products>
    <Product Id = "01" Name = "Shoes" Brand = "Nike" />
    <Product Id = "02" Name = "Cap" Brand = "Adidas" />
    <Product Id = "03" Name = "Shirt" Brand = "Raymond" />
    <Product Id = "04" Name = "Watch" Brand = "Titan" />
    <Product Id = "05" Name = "Mobile" Brand = "Apple" />
    <Product Id = "06" Name = "Laptop" Brand = "Microsoft" />
    <Product Id = "07" Name = "Camera" Brand = "Canon" />
    <Product Id = "08" Name = "Bike" Brand = "BMW" />
    <Product Id = "09" Name = "Car" Brand = "Audi" />
    <Product Id = "10" Name = "HeadPhones" Brand = "Sony" />
</Products>
```

Output:

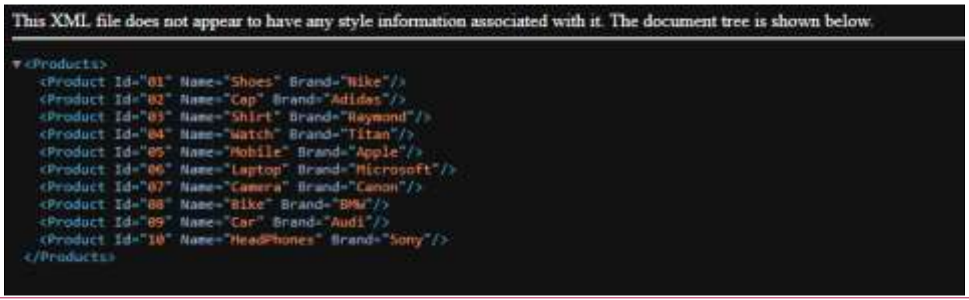
TagBased XML:



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<Products>
  ▼<Product>
     <Id>01</Id>
     <Name>Shoes</Name>
     <Brand>Nike</Brand>
   </Product>
  ▼<Product>
     <Id>02</Id>
     <Name>Cap</Name>
     <Brand>Adidas</Brand>
   </Product>
  ▼<Product>
     <Id>03</Id>
     <Name>Shirt</Name>
     <Brand>Raymond</Brand>
   </Product>
  ▼<Product>
     <Id>04</Id>
     <Name>Watch</Name>
     <Brand>Titan</Brand>
   </Product>
  ▼<Product>
     <Id>05</Id>
     <Name>Mobile</Name>
     <Brand>Apple</Brand>
   </Product>
  ▼<Product>
     <Id>06</Id>
     <Name>Laptop</Name>
     <Brand>Microsoft</Brand>
   </Product>
  ▼<Product>
     <Id>07</Id>
     <Name>Camera</Name>
     <Brand>Canon</Brand>
   </Product>
  ▼<Product>
     <Id>08</Id>
     <Name>Bike</Name>
     <Brand>BMW</Brand>
   </Product>
  ▼<Product>
     <Id>09</Id>
     <Name>Car</Name>
     <Brand>Audi</Brand>
   </Product>
  ▼<Product>
     <Id>10</Id>
     <Name>HeadPhones</Name>
     <Brand>Sony</Brand>
   </Product>
 </Products>
```

AttributeBased XML:



This XML file does not appear to have any style information associated with it. The document tree is shown below.

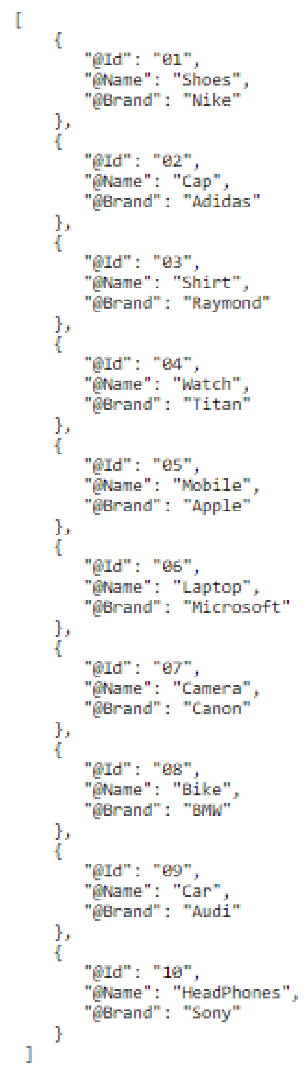
```
▼<Products>
   <Product Id="01" Name="Shoes" Brand="Nike"/>
   <Product Id="02" Name="Cap" Brand="Adidas"/>
   <Product Id="03" Name="Shirt" Brand="Raymond"/>
   <Product Id="04" Name="Watch" Brand="Titan"/>
   <Product Id="05" Name="Mobile" Brand="Apple"/>
   <Product Id="06" Name="Laptop" Brand="Microsoft"/>
   <Product Id="07" Name="Camera" Brand="Canon"/>
   <Product Id="08" Name="Bike" Brand="BMW"/>
   <Product Id="09" Name="Car" Brand="Audi"/>
   <Product Id="10" Name="HeadPhones" Brand="Sony"/>
 </Products>
```

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

Code:

```json
[
    {
        "@Id": "01",
        "@Name": "Shoes",
        "@Brand": "Nike"
    },
    {
        "@Id": "02",
        "@Name": "Cap",
        "@Brand": "Adidas"
    },
    {
        "@Id": "03",
        "@Name": "Shirt",
        "@Brand": "Raymond"
    },
    {
        "@Id": "04",
        "@Name": "Watch",
        "@Brand": "Titan"
    },
    {
        "@Id": "05",
        "@Name": "Mobile",
        "@Brand": "Apple"
    },
    {
        "@Id": "06",
        "@Name": "Laptop",
        "@Brand": "Microsoft"
    },
    {
        "@Id": "07",
        "@Name": "Camera",
        "@Brand": "Canon"
    },
    {
        "@Id": "08",
        "@Name": "Bike",
        "@Brand": "BMW"
    },
    {
        "@Id": "09",
        "@Name": "Car",
        "@Brand": "Audi"
    },
    {
        "@Id": "10",
        "@Name": "HeadPhones",
        "@Brand": "Sony"
    }
]
```

Output:



```
[
    {
        "@Id": "01",
        "@Name": "Shoes",
        "@Brand": "Nike"
    },
    {
        "@Id": "02",
        "@Name": "Cap",
        "@Brand": "Adidas"
    },
    {
        "@Id": "03",
        "@Name": "Shirt",
        "@Brand": "Raymond"
    },
    {
        "@Id": "04",
        "@Name": "Watch",
        "@Brand": "Titan"
    },
    {
        "@Id": "05",
        "@Name": "Mobile",
        "@Brand": "Apple"
    },
    {
        "@Id": "06",
        "@Name": "Laptop",
        "@Brand": "Microsoft"
    },
    {
        "@Id": "07",
        "@Name": "Camera",
        "@Brand": "Canon"
    },
    {
        "@Id": "08",
        "@Name": "Bike",
        "@Brand": "BMW"
    },
    {
        "@Id": "09",
        "@Name": "Car",
        "@Brand": "Audi"
    },
    {
        "@Id": "10",
        "@Name": "HeadPhones",
        "@Brand": "Sony"
    }
]
```

5. Write the benefits of JSON over XML

- JSON occupy less memory than XML.
- JSON is much easier to parse.
- JSON has lower character count.

6. Factorial of a number.

Code:

Mathematics:

```
namespace MatheMatics
{
    public class Algebra
    {
        public 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;
            }
        }
    }
}
```

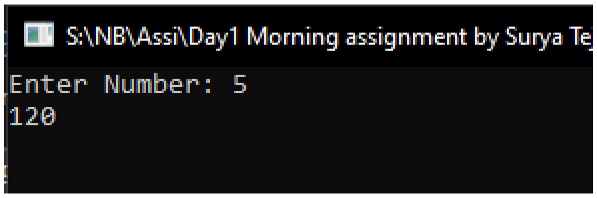
ConsoleApplication:

```
using System;
using MatheMatics;

namespace ConsoleApplication
{
    public class Program
    {
        static void Main(string[] args)
        {
            int n;
            Algebra al = new Algebra();

            Console.Write("Enter Number: ");
            n = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine(al.Factorial(n));

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

WindowsApplication:

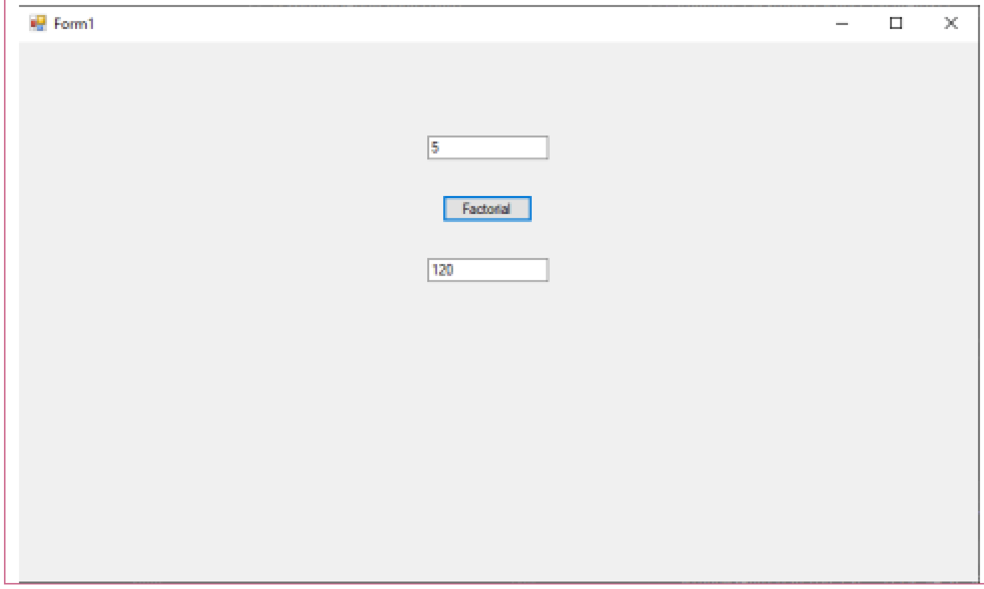
```
using System;
using System.Windows.Forms;
using MatheMatics;

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

        private void button1_Click(object sender, EventArgs e)
        {
            Algebra all = new Algebra();

            int n = Convert.ToInt32(textBox1.Text);
            int fact = all.Factorial(n);

            textBox2.Text = fact.ToString();

        }
    }
}
```

Output:



S:\NB\Assi\Day1 Morning assignment by Surya Te

```
Enter Number: 5
120
```

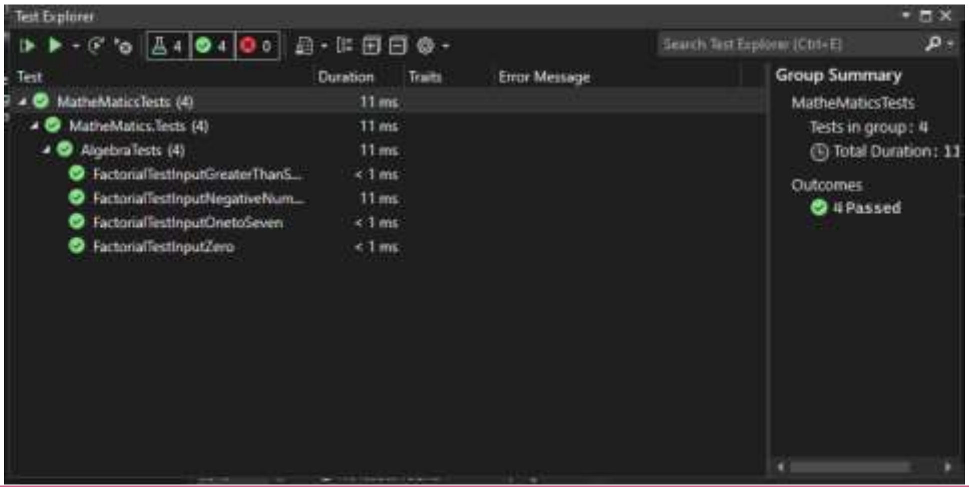


7. Implement TDD and write 4 test cases

Code:

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using MatheMatics;
using System;

namespace MatheMatics.Tests
{
    [TestClass()]
    public class AlgebraTests
    {
        Algebra a = new Algebra();
        [TestMethod()]
        public void FactorialTestInputZero()
        {
            //Arrange
            int n = 0, actual, expected = 1;
            //Act
            actual = a.Factorial(n);
            //Assert
            Assert.AreEqual(expected, actual);
        }
        [TestMethod()]
        public void FactorialTestInputOnetoSeven()
        {
            //Arrange
            int n = 5, actual, expected = 120;
            //Act
            actual = a.Factorial(n);
            //Assert
            Assert.AreEqual(expected, actual);
        }
        [TestMethod()]
        public void FactorialTestInputNegativeNumbers()
        {
            //Arrange
            int n = -7, actual, expected = -9999;
            //Act
            actual = a.Factorial(n);
            //Assert
            Assert.AreEqual(expected, actual);
        }
        [TestMethod()]
        public void FactorialTestInputGreaterThanSeven()
        {
            //Arrange
            int n = 9, actual, expected = -999;
            //Act
            actual = a.Factorial(n);
            //Assert
            Assert.AreEqual(expected, actual);
        }
    }

}
```

Output:



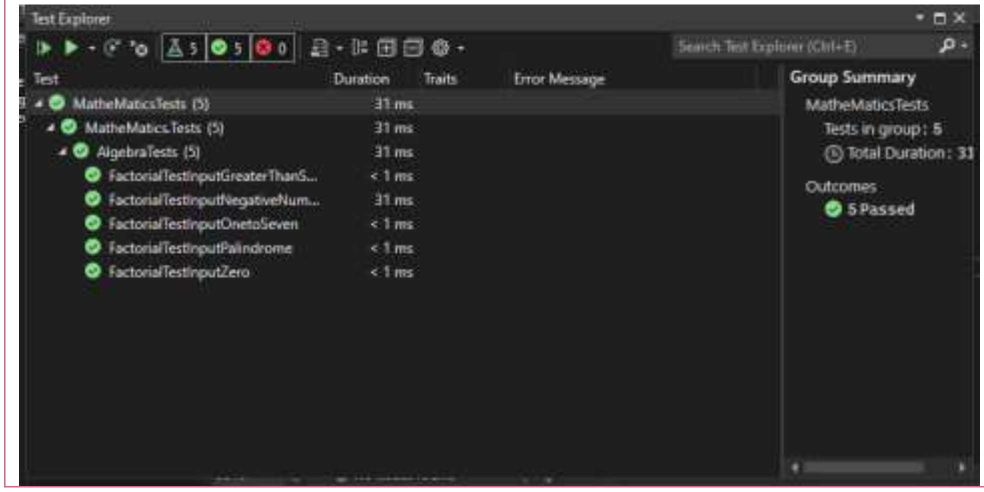
```
Test                                   Duration   Traits   Error Message      Group Summary
MatheMaticsTests (4)                   11 ms                                    MatheMaticsTests
  MatheMatics.Tests (4)                11 ms                                      Tests in group : 4
    AlgebraTests (4)                   11 ms                                      Total Duration : 11
      FactorialTestInputGreaterThanS... < 1 ms
      FactorialTestInputNegativeNum...  11 ms                                   Outcomes
      FactorialTestInputOnetoSeven      < 1 ms                                    4 Passed
      FactorialTestInputZero            < 1 ms
```

8. Check if the number is palindrome and Implement TDD.

Code:

```
public int Palindrome(int num)
        {
            int rem, temp, sum = 0;

            temp = num;
            while (num > 0)
            {
                rem = num % 10;
                sum = (sum * 10) + rem;
                num = num / 10;
            }
            if (temp == sum)
                Console.Write($"{temp} is Palindrome.");
            else
                Console.Write($"{temp} is not Palindrome");
            return temp;

        }
```

Output:



```
Test Explorer
5  5  0                                    Search Test Explorer (Ctrl+E)
Test                                 Duration   Traits   Error Message    Group Summary
MatheMaticsTests (5)                   31 ms                               MatheMaticsTests
  MatheMatics.Tests (5)                31 ms                                 Tests in group: 5
    AlgebraTests (5)                   31 ms                                 Total Duration: 31
      FactorialTestInputGreaterThanS...  < 1 ms                            Outcomes
      FactorialTestInputNegativeNum...   31 ms                               5 Passed
      FactorialTestInputOnetoSeven       < 1 ms
      FactorialTestInputPalindrome       < 1 ms
      FactorialTestInputZero             < 1 ms
```