Date : 23-08-24

1.
Code :

```
using System.Xml;

namespace Ex1
{

    class Assets
    {
        public int AssetID { get; set; }
        public string AssetName { get; set; }

        public Assets(int assetID,string assetName)
        {
            AssetID = assetID;
            AssetName = assetName;
        }

    }
    internal class Program
    {
        private static void Main(string[] args)
        {

            Assets[] assets = new Assets[3];
            assets[0] = new Assets(1, "Laptop");
            assets[1] = new Assets(2, "Mouse");
            assets[2] = new Assets(3, "Printer");
            using (XmlWriter writer = XmlWriter.Create("Assets.xml"))
            {
                writer.WriteStartDocument();
                writer.WriteStartElement("Assets");
                foreach (Assets asset in assets)
                {
```

```
                writer.WriteStartElement("Assets");
                writer.WriteElementString("AssetID", asset.AssetID.ToString());
                writer.WriteElementString("AssetName", asset.AssetName.ToString());
                writer.WriteEndElement();
            }
            writer.WriteEndElement();
            writer.WriteEndDocument();
        }

    }
  }

}
```

Output :

Assets.xml

2.

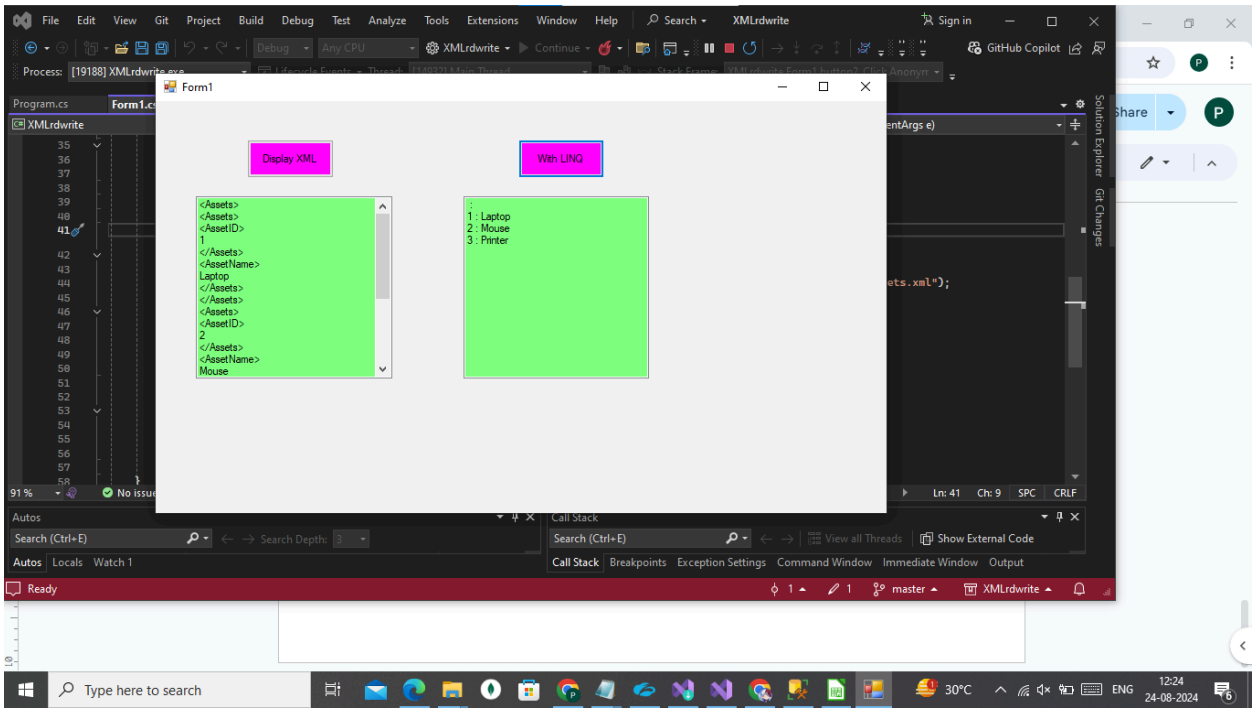
Code :

```
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 System.Xml;
using System.Xml.Linq;

namespace XMLrdwrite
{
  public partial class Form1 : Form
  {
```

```
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            XmlReader xmlread =
XmlReader.Create(@"C:\Users\Prathik.b\Source\Repos\XMLReadWrite\XMLReadWrite\
bin\Debug\net8.0\Assets.xml");
            while (xmlread.Read())
            {
                switch (xmlread.NodeType)
                {
                    case XmlNodeType.Element:
                        listBox1.Items.Add("<" + xmlread.Name + ">");
                        break;
                    case XmlNodeType.Text:
                        listBox1.Items.Add(xmlread.Value);
                        break;
                    case XmlNodeType.EndElement:
                        listBox1.Items.Add("</Assets>");
                        break;
                }
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            XDocument doc =
XDocument.Load(@"C:\Users\Prathik.b\Source\Repos\XMLReadWrite\XMLReadWrite\bi
n\Debug\net8.0\Assets.xml");
            var elets = from ele in doc.Descendants("Assets")
                        select new
                        {
                            AssetId = (string)ele.Element("AssetID"),
```

```
                AssetName = (string)ele.Element("AssetName")
            };


        foreach (var el in elets)
        {
            listBox2.Items.Add(el.AssetId + " : " + el.AssetName);
        }
    }
  }
}
```

Output :



3.

Code : Test App

```
namespace SampleTest
{
   public class MathOpe
   {
      public int add(int a, int b)
      {
         return a + b;
      }
      public int Sub(int x, int y)
      {
         return x - y;
      }
      public int Pro(int x, int y)
      {
         return x * y;
      }
      public int Div(int x, int y)
      {
         return x / y;
      }
      public virtual bool CheckValues()
      {
         return false;
      }
   }
   public class Employee
   {
      string Name;
      int Age;
      public Employee(string nme, int age)
      {
         Name = nme;
         Age = age;
      }
```

```
        public string name
        {
            get
            {
                return Name;
            }
            set
            {
                Name = value;
            }
        }
        public int age
        {
            get
            {
                return Age;
            }
            set { Age = value; }
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

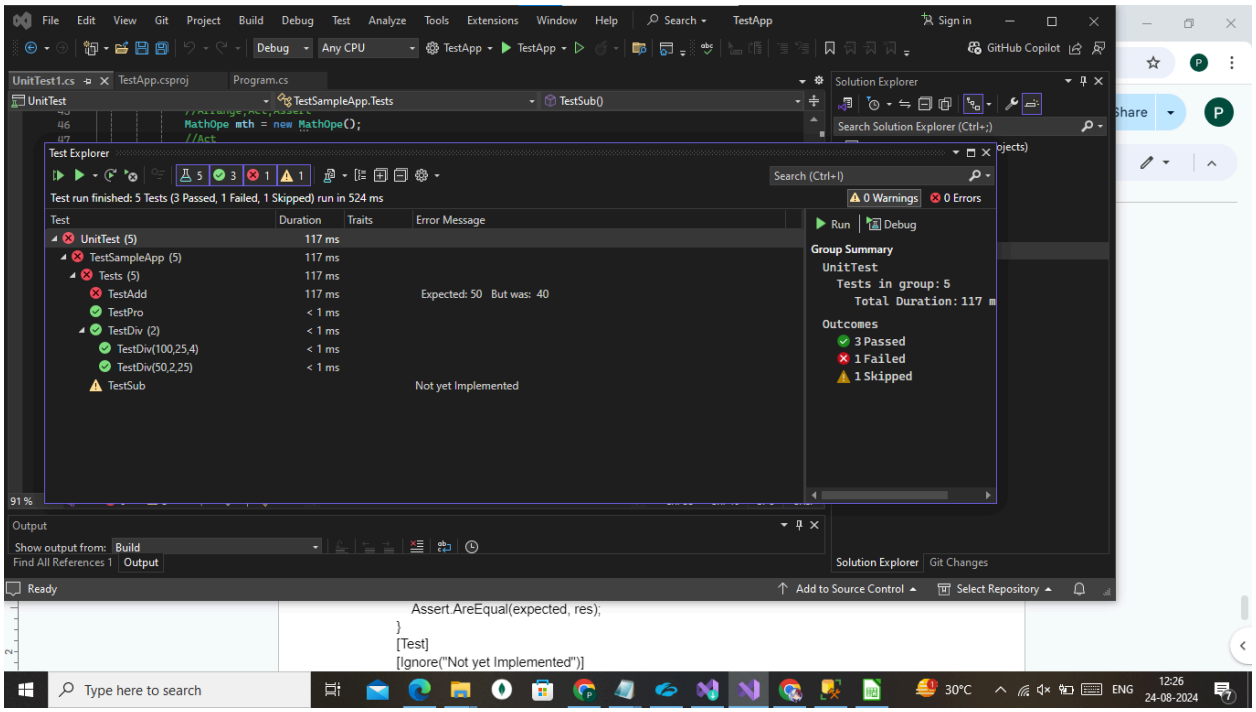
Code : Unit Test

```
using SampleTest;

namespace TestSampleApp
```

```
{
    public class Tests
    {

        public int i = 50, j = 20;
        public bool result;
        [SetUp] //performed initially
        public void CheckNonNegative()
        {
            if (i > 0 && j > 0)
            {
                result = true;
            }
            else
            {
                result = false;
            }
        }
        [Test] //TestMethod
        public void TestAdd()
        {
            if (result)
            {
                //Arrange,- creating istance for the ttest method
                MathOpe mth = new MathOpe();
                //Act - calling the method by passing parameter
                var res = mth.add(20, 20);//actual value
                                    //Assert
                Assert.AreEqual(50, res);
            }
            else
            {
                Assert.Fail();
            }

        }
```

```
        [Test]
        [TestCase(100, 25, 4)]
        [TestCase(50, 2, 25)]
        public void TestDiv(int a, int b, int expected)
        {
            //Arrange,Act,Assert
            MathOpe mth = new MathOpe();
            //Act
            var res = mth.Div(a, b);
            //Assert
            Assert.AreEqual(expected, res);
        }
        [Test]
        [Ignore("Not yet Implemented")]
        public void TestSub()
        {

        }
        [Test]
        public void TestPro()
        {
            //Arrange,Act,Assert
            MathOpe mth = new MathOpe();
            //Act
            var res = mth.Pro(i, j);
            //Assert
            Assert.AreEqual(1000, res);
        }

    }
}
```

Output :


Test run finished: 5 Tests (3 Passed, 1 Failed, 1 Skipped) run in 524 ms
Test
Duration
Traits
Error Message
UnitTest (5) 117 ms
TestSampleApp (5) 117 ms
Tests (5) 117 ms
TestAdd 117 ms Expected: 50 But was: 40
TestPro < 1 ms
TestDiv (2) < 1 ms
TestDiv(100,25,4) < 1 ms
TestDiv(50,2,25) < 1 ms
TestSub Not yet Implemented
Group Summary
UnitTest
Tests in group: 5
Outcomes
3 Passed
1 Failed
1 Skipped
Assert.AreEqual(expected, res);
}
[Test]
[Ignore("Not yet Implemented")]