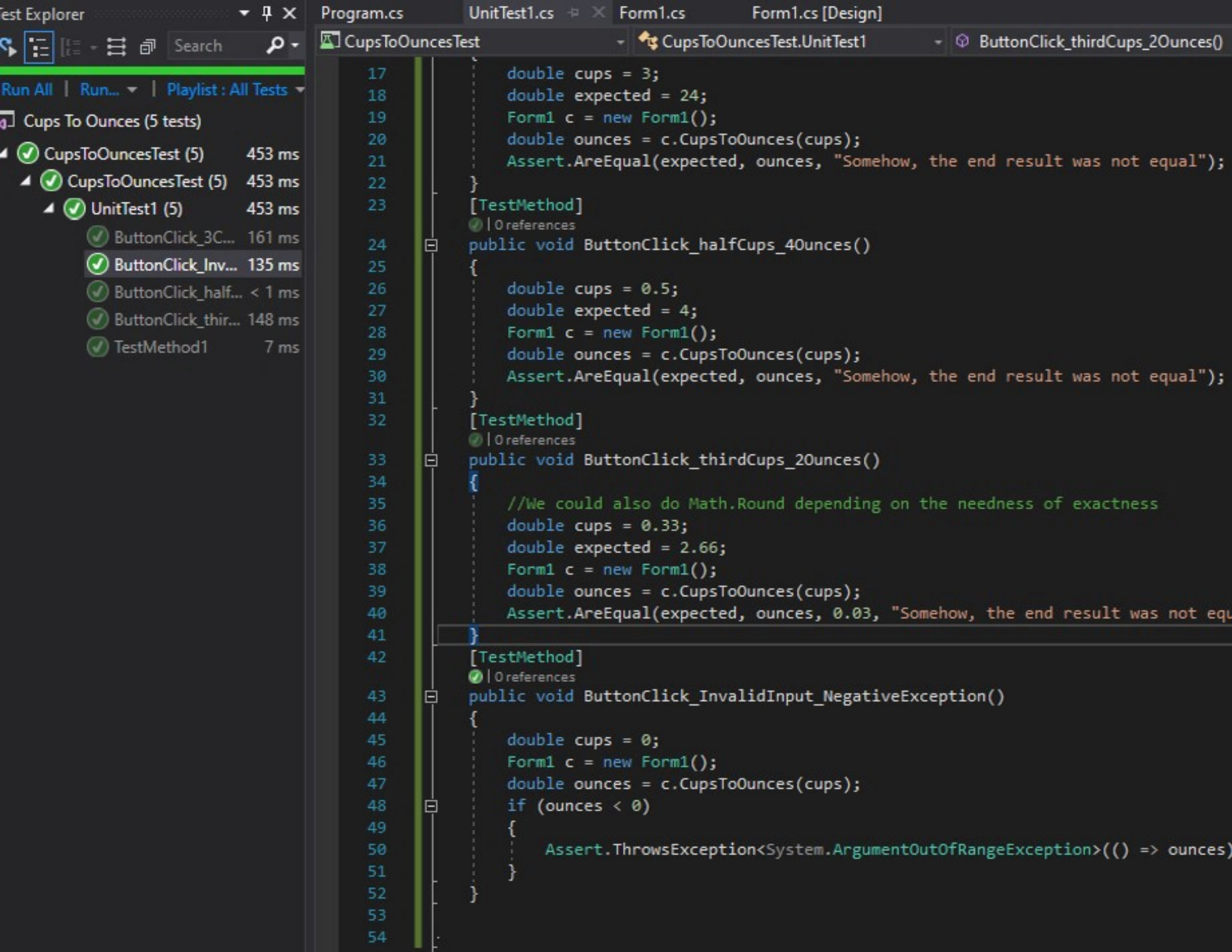

Test Explorer
Search
Run All | Run... | Playlist : All Tests
Cups To Ounces (5 tests)
CupsToOuncesTest (5) 453 ms
CupsToOuncesTest (5) 453 ms
UnitTest1 (5) 453 ms
ButtonClick_3C... 161 ms
ButtonClick_Inv... 135 ms
ButtonClick_half... < 1 ms
ButtonClick_thir... 148 ms
TestMethod1 7 ms
Program.cs
UnitTest1.cs
Form1.cs
Form1.cs [Design]
CupsToOuncesTest
CupsToOuncesTest.UnitTest1
ButtonClick_thirdCups_2Ounces()
17 double cups = 3;
18 double expected = 24;
19 Form1 c = new Form1();
20 double ounces = c.CupsToOunces(cups);
21 Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
22 }
23 [TestMethod]
0 references
24 public void ButtonClick_halfCups_4Ounces()
25 {
26 double cups = 0.5;
27 double expected = 4;
28 Form1 c = new Form1();
29 double ounces = c.CupsToOunces(cups);
30 Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
31 }
32 [TestMethod]
0 references
33 public void ButtonClick_thirdCups_2Ounces()
34 {
35 //We could also do Math.Round depending on the needness of exactness
36 double cups = 0.33;
37 double expected = 2.66;
38 Form1 c = new Form1();
39 double ounces = c.CupsToOunces(cups);
40 Assert.AreEqual(expected, ounces, 0.03, "Somehow, the end result was not equ
41 }
42 [TestMethod]
0 references
43 public void ButtonClick_InvalidInput_NegativeException()
44 {
45 double cups = 0;
46 Form1 c = new Form1();
47 double ounces = c.CupsToOunces(cups);
48 if (ounces < 0)
49 {
50 Assert.ThrowsException<System.ArgumentOutOfRangeException>(() => ounces)
51 }
52 }
53
54

```
ng System;
ng System.Windows.Forms;
ng Microsoft.VisualStudio.TestTools.UnitTesting;
ng Cups_To_Ounces;
espace CupsToOuncesTest

 [TestClass]
 0 references
 public class UnitTest1
 {
     [TestMethod]
     0 references
     public void TestMethod1()
     {
     }
     [TestMethod]
     0 references
     public void ButtonClick_3Cups_24Ounces()
     {
         double cups = 3;
         double expected = 24;
         Form1 c = new Form1();
         double ounces = c.CupsToOunces(cups);
         Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal
     }
 }
```

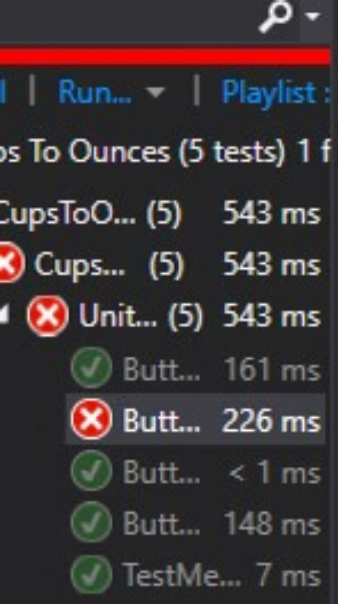
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Cups_To_Ounces;
namespace CupsToOuncesTest
{
    [TestClass]
    0 references
    public class UnitTest1
    {
        [TestMethod]
        0 references
        public void TestMethod1()
        {
        }
        [TestMethod]
        0 references
        public void ButtonClick_3Cups_24Ounces()
        {
            double cups = 3;
            double expected = 24;
            Form1 c = new Form1();
            double ounces = c.CupsToOunces(cups);
            Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
        }
        [TestMethod]
        0 references
        public void ButtonClick_halfCups_4Ounces()
        {
            double cups = 0.5;
            double expected = 4;
            Form1 c = new Form1();
            double ounces = c.CupsToOunces(cups);
            Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
        }

    }
```

Program.cs | UnitTest1.cs* | Form1.cs | Form1.cs [Design]

CupsToOuncesTest | CupsToOuncesTest.UnitTest1 | ButtonClick_thirdC

```
16          {
17              double cups = 3;
18              double expected = 24;
19              Form1 c = new Form1();
20              double ounces = c.CupsToOunces(cups);
21              Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
22          }
23          [TestMethod]
                0 references
24          public void ButtonClick_halfCups_4Ounces()
25          {
26              double cups = 0.5;
27              double expected = 4;
28              Form1 c = new Form1();
29              double ounces = c.CupsToOunces(cups);
30              Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
31          }
32          [TestMethod]
                0 references
33          public void ButtonClick_thirdCups_2Ounces()
34          {
35              //We could also do Math.Round depending on the needness of exactness
36              double cups = 0.33;
37              double expected = 2.66;
38              Form1 c = new Form1();
39              double ounces = c.CupsToOunces(cups);
40              Assert.AreEqual(expected, ounces, 0.03, "Somehow, the end result was not equal");
41          }
42
43      }
44
45
```




```
18          double expected = 24;
19          Form1 c = new Form1();
20          double ounces = c.CupsToOunces(cups);
21          Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
22      }
23      [TestMethod]
            0 references
24      public void ButtonClick_halfCups_4Ounces()
25      {
26          double cups = 0.5;
27          double expected = 4;
28          Form1 c = new Form1();
29          double ounces = c.CupsToOunces(cups);
30          Assert.AreEqual(expected, ounces, "Somehow, the end result was not equal");
31      }
32      [TestMethod]
            0 references
33      public void ButtonClick_thirdCups_2Ounces()
34      {
35          //We could also do Math.Round depending on the needness of exactness
36          double cups = 0.33;
37          double expected = 2.66;
38          Form1 c = new Form1();
39          double ounces = c.CupsToOunces(cups);
40          Assert.AreEqual(expected, ounces, 0.03, "Somehow, the end result was not equal
41      }
42      [TestMethod]
            0 references
43      public void ButtonClick_InvalidInput_NegativeException()
44      {
45          double cups = 0;
46          Form1 c = new Form1();
47          double ounces = c.CupsToOunces(cups);
48          Assert.ThrowsException<System.ArgumentOutOfRangeException>(() => ounces);
49      }
50
51
52
53
```