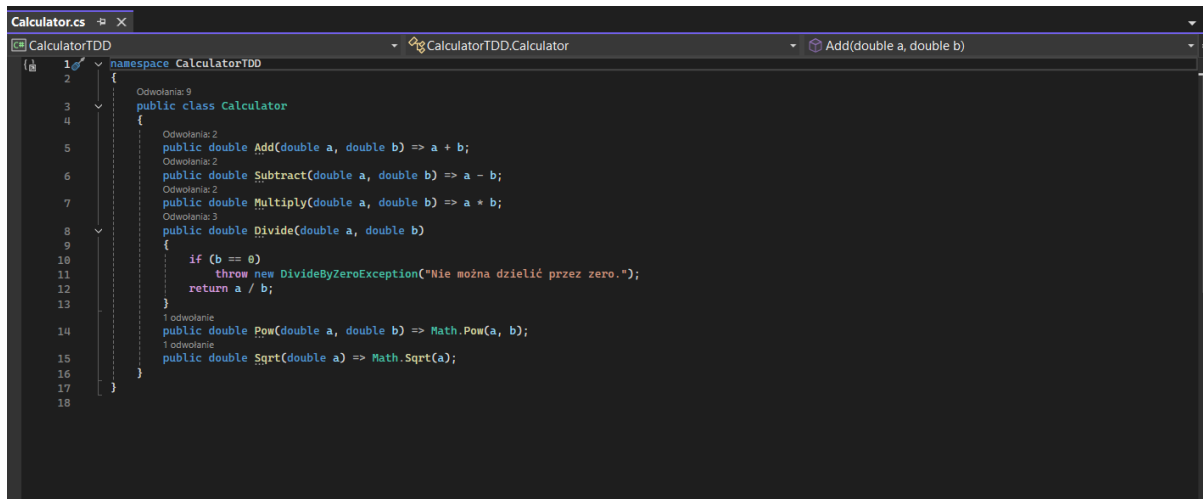


SPRAWOZDANIE

Kalkulator TDD

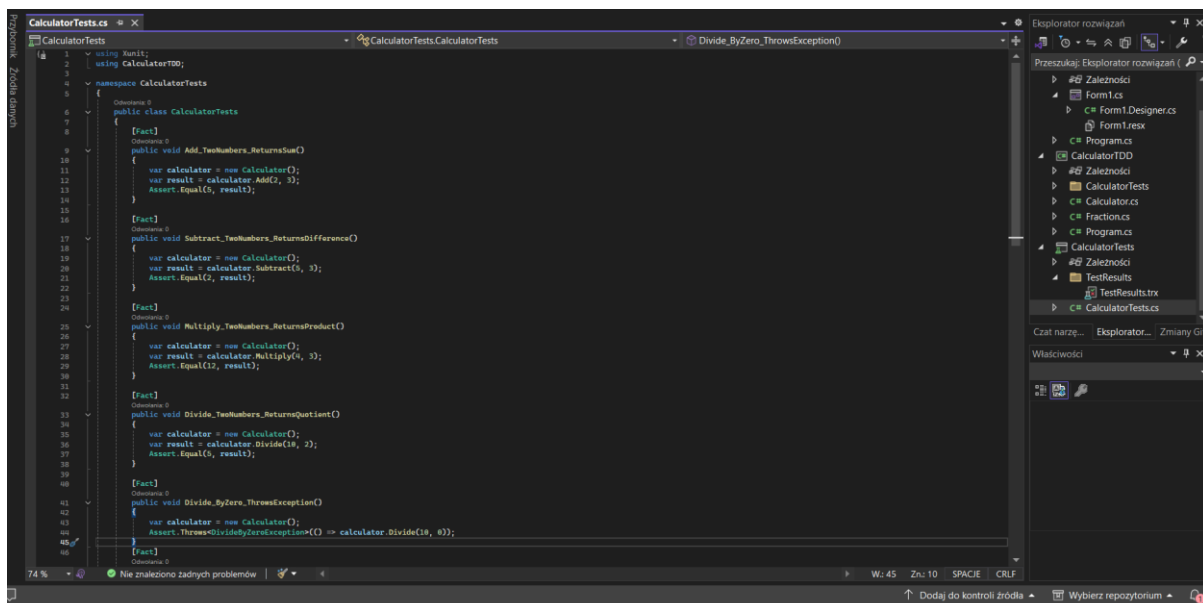
Autor: Patryk Figas

1. Klasa Calculator posiada 6 metod:

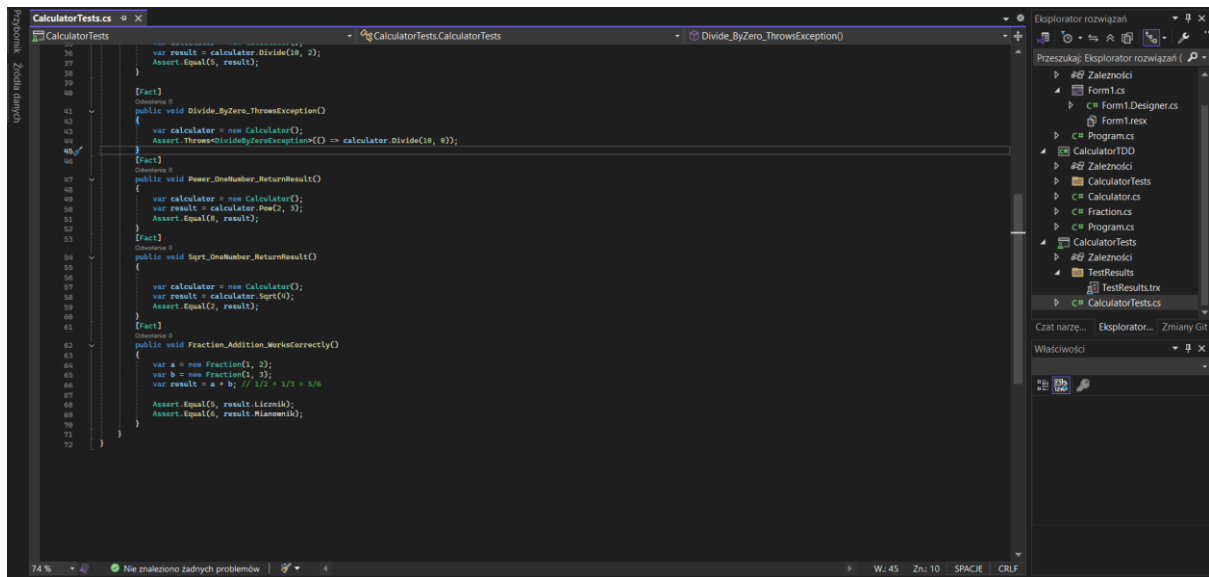


```
1 namespace CalculatorTDD
2 {
3     public class Calculator
4     {
5         Odwołania: 2
6         public double Add(double a, double b) => a + b;
7         Odwołania: 2
8         public double Subtract(double a, double b) => a - b;
9         Odwołania: 2
10        public double Multiply(double a, double b) => a * b;
11        Odwołania: 3
12        public double Divide(double a, double b)
13        {
14            if (b == 0)
15            {
16                throw new DivideByZeroException("Nie można dzielić przez zero.");
17            }
18            return a / b;
19        }
20        Odwołania: 1
21        public double Pow(double a, double b) => Math.Pow(a, b);
22        Odwołania: 1
23        public double Sqrt(double a) => Math.Sqrt(a);
24    }
25 }
```

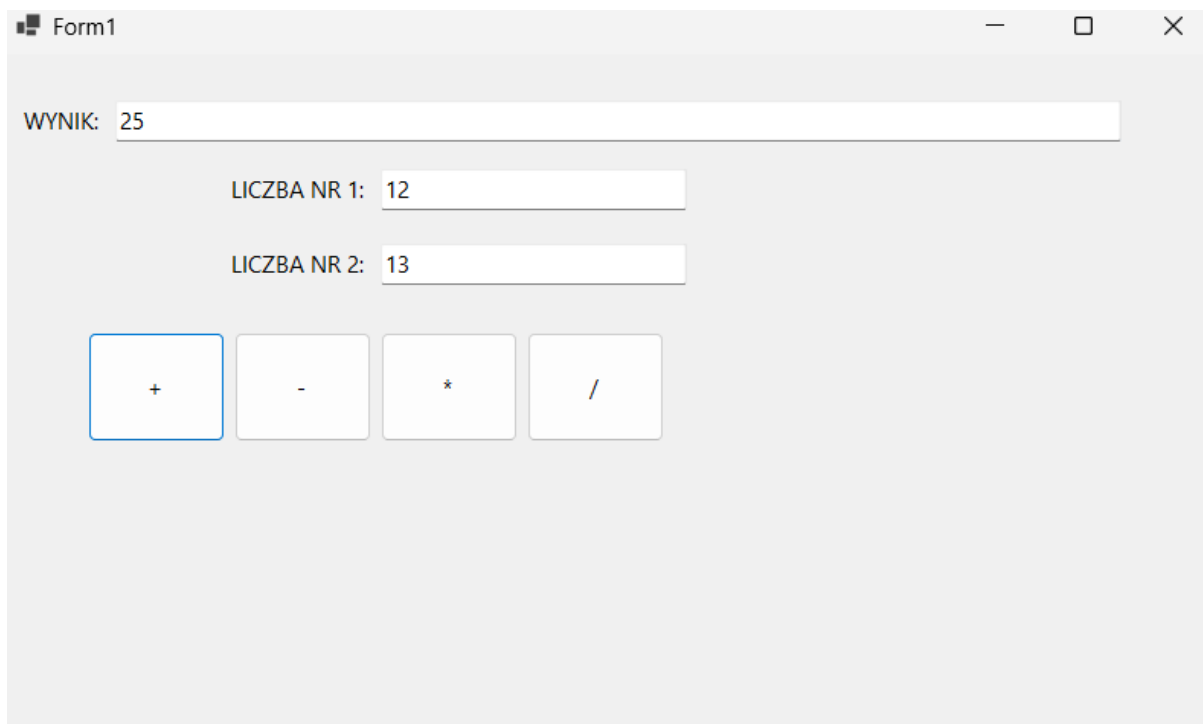
2. Każda metoda ma swój własny test jednostkowy:



```
1 using Xunit;
2 using CalculatorTDD;
3
4 namespace CalculatorTests
5 {
6     public class CalculatorTests
7     {
8         [Fact]
9         public void Add_TwoNumbers_ReturnsSum()
10        {
11            var calculator = new Calculator();
12            var result = calculator.Add(2, 3);
13            Assert.Equal(5, result);
14        }
15
16        [Fact]
17        public void Subtract_TwoNumbers_ReturnsDifference()
18        {
19            var calculator = new Calculator();
20            var result = calculator.Subtract(5, 3);
21            Assert.Equal(2, result);
22        }
23
24        [Fact]
25        public void Multiply_TwoNumbers_ReturnsProduct()
26        {
27            var calculator = new Calculator();
28            var result = calculator.Multiply(4, 3);
29            Assert.Equal(12, result);
30        }
31
32        [Fact]
33        public void Divide_TwoNumbers_ReturnsQuotient()
34        {
35            var calculator = new Calculator();
36            var result = calculator.Divide(10, 2);
37            Assert.Equal(5, result);
38        }
39
40        [Fact]
41        public void Divide_ByZero_ThrowsException()
42        {
43            var calculator = new Calculator();
44            Assert.Throws<DivideByZeroException>(() => calculator.Divide(10, 0));
45        }
46    }
47 }
```



4. Stworzyłem do tego proste GUI obsługujące 4 podstawowe metody:



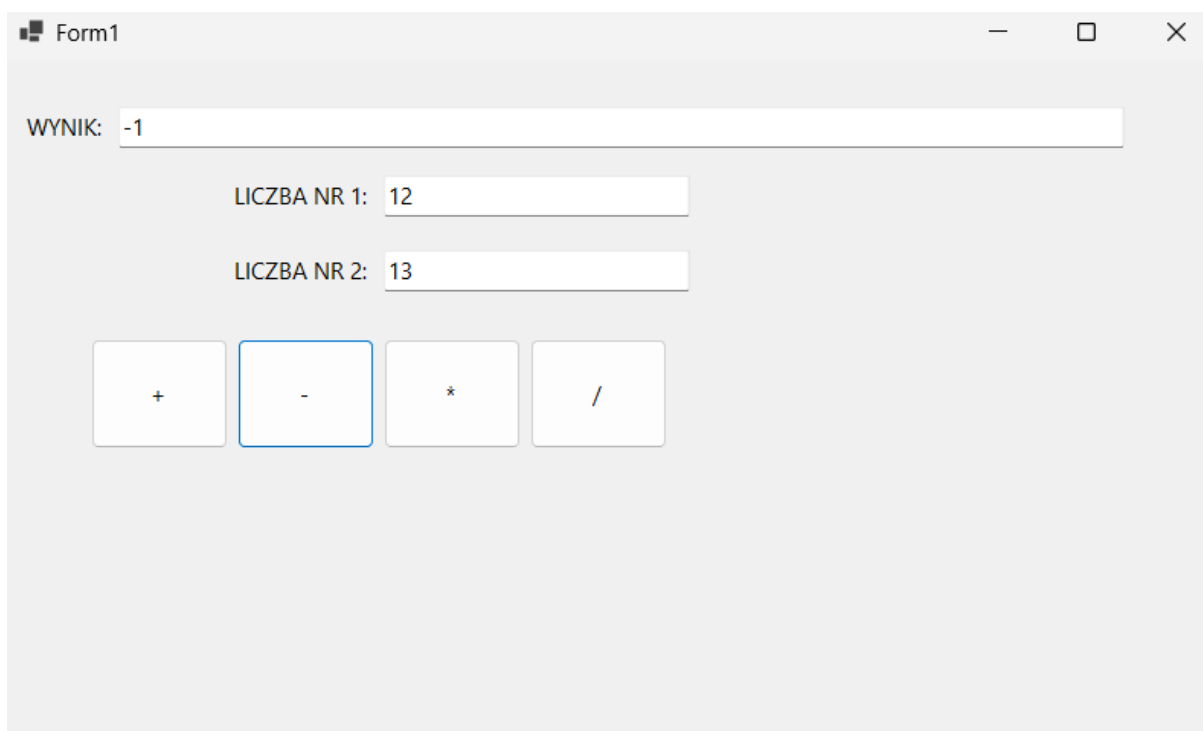
Form1

WYNIK: 25

LICZBA NR 1: 12

LICZBA NR 2: 13

+ - * /



Form1

WYNIK: -1

LICZBA NR 1: 12

LICZBA NR 2: 13

+ - * /

Form1

WYNIK: 156

LICZBA NR 1: 12

LICZBA NR 2: 13

+ - * /

Form1

WYNIK: 0,9230769230769231

LICZBA NR 1: 12

LICZBA NR 2: 13

+ - * /