COVER PAGE

Name: Eric Gathinji

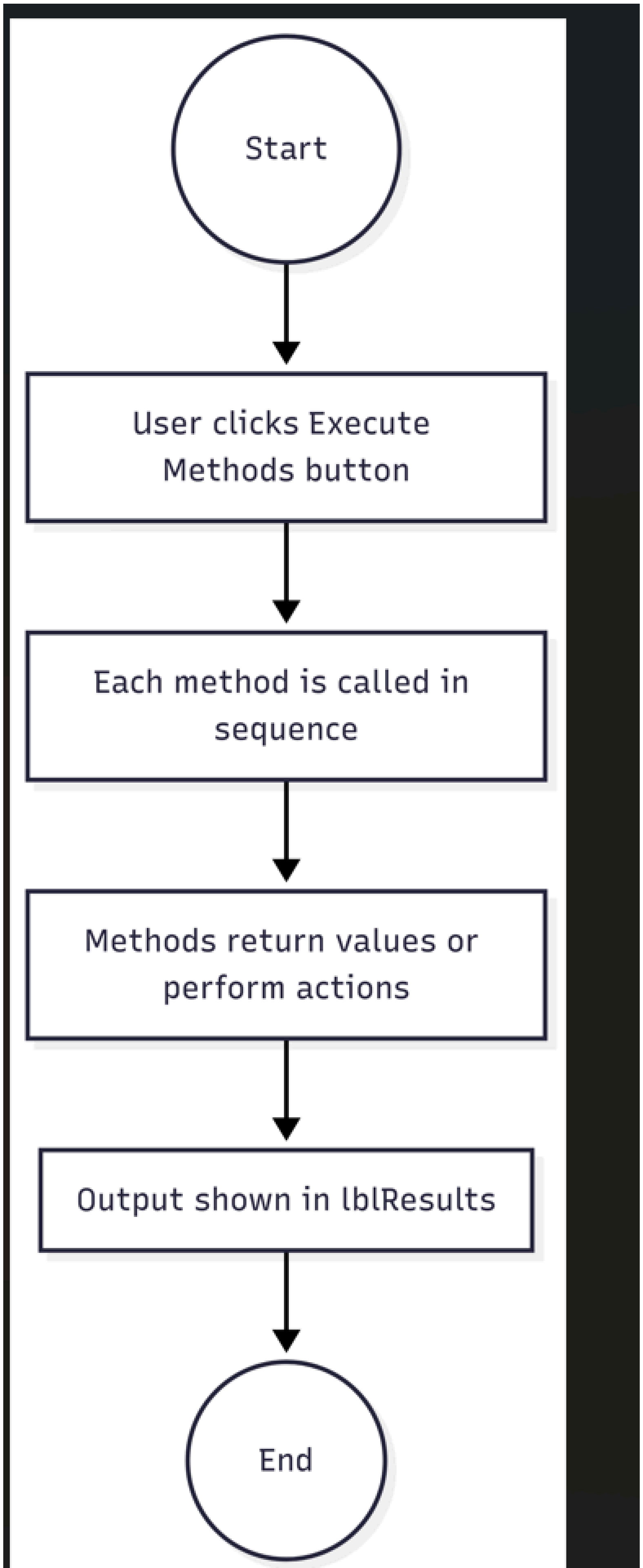
Programming in C# CST-150-0500

Grand Canyon University

5th July 2025

Activity 4 Part 2

FLOWCHART

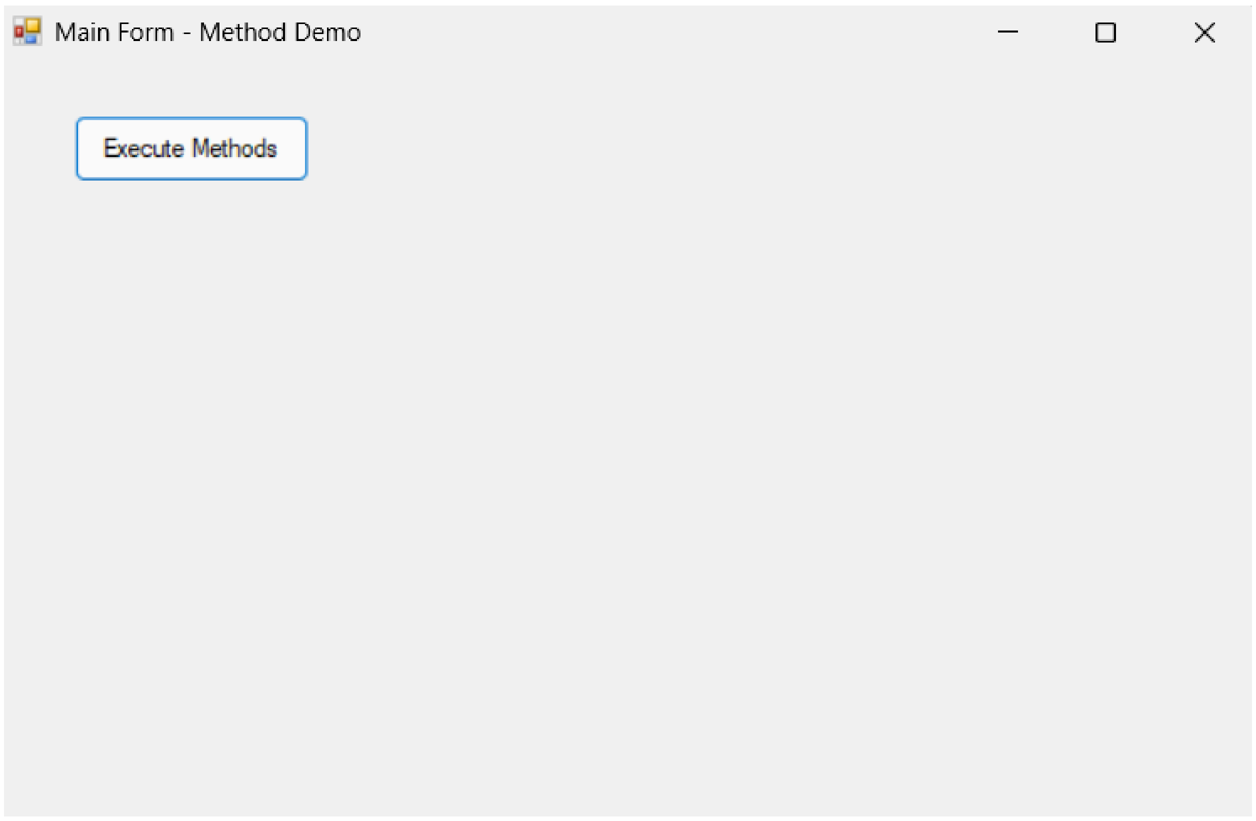

Start
User clicks Execute Methods button
Each method is called in sequence
Methods return values or perform actions
Output shown in lblResults
End

Flowchart showing sequence of method execution upon button click.

EXPLANATION;

This flowchart outlines the logic of the Part 2 program.
A single button triggers nine different methods, each performing a specific operation.
Results are displayed in a label sequentially to demonstrate control flow and output.

Screenshot of the Form Before Populated

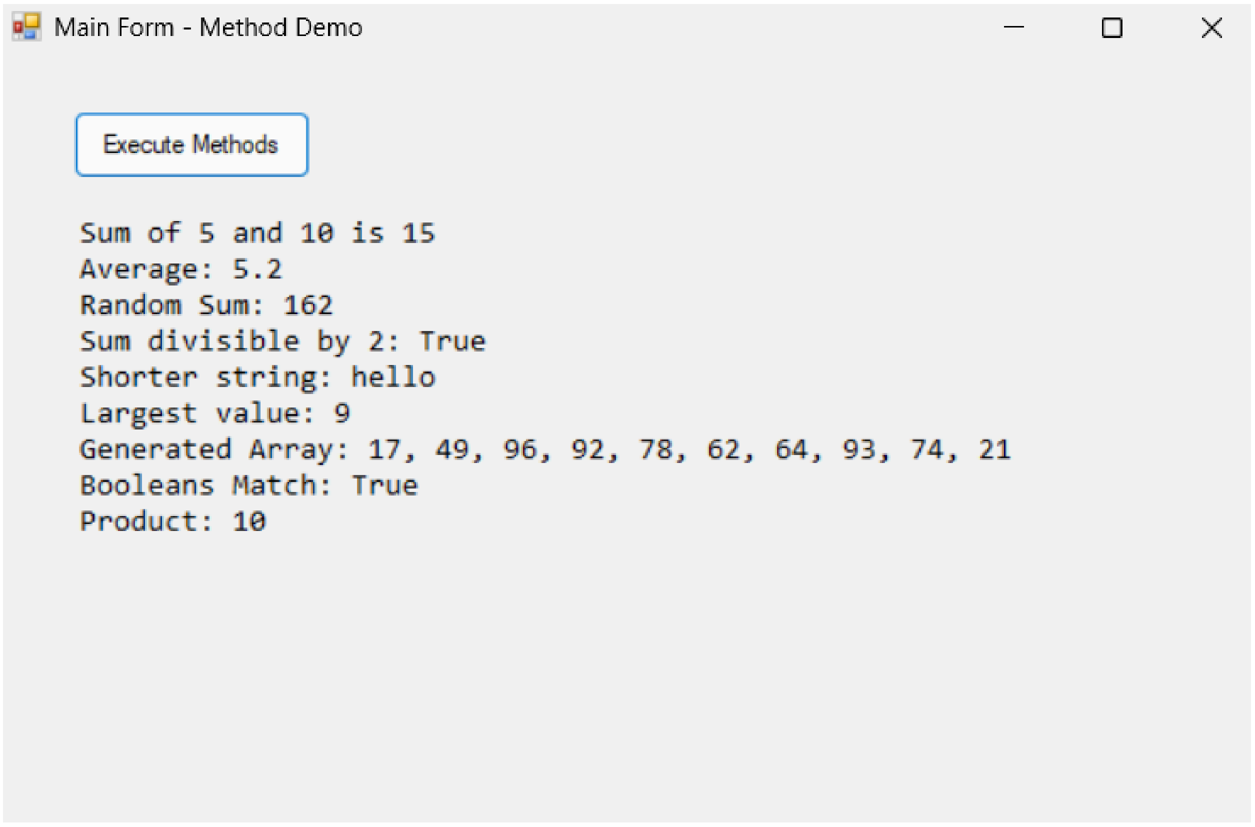


Form before user interaction

EXPLANATION;

This is the initial state of the form.
It contains a button labeled “Execute Methods” and a results label that is initially empty.
The form layout is clean and simple to support method demonstration.

SCREENSHOTS OF THE FORMS AFTER BEING POPULATED;

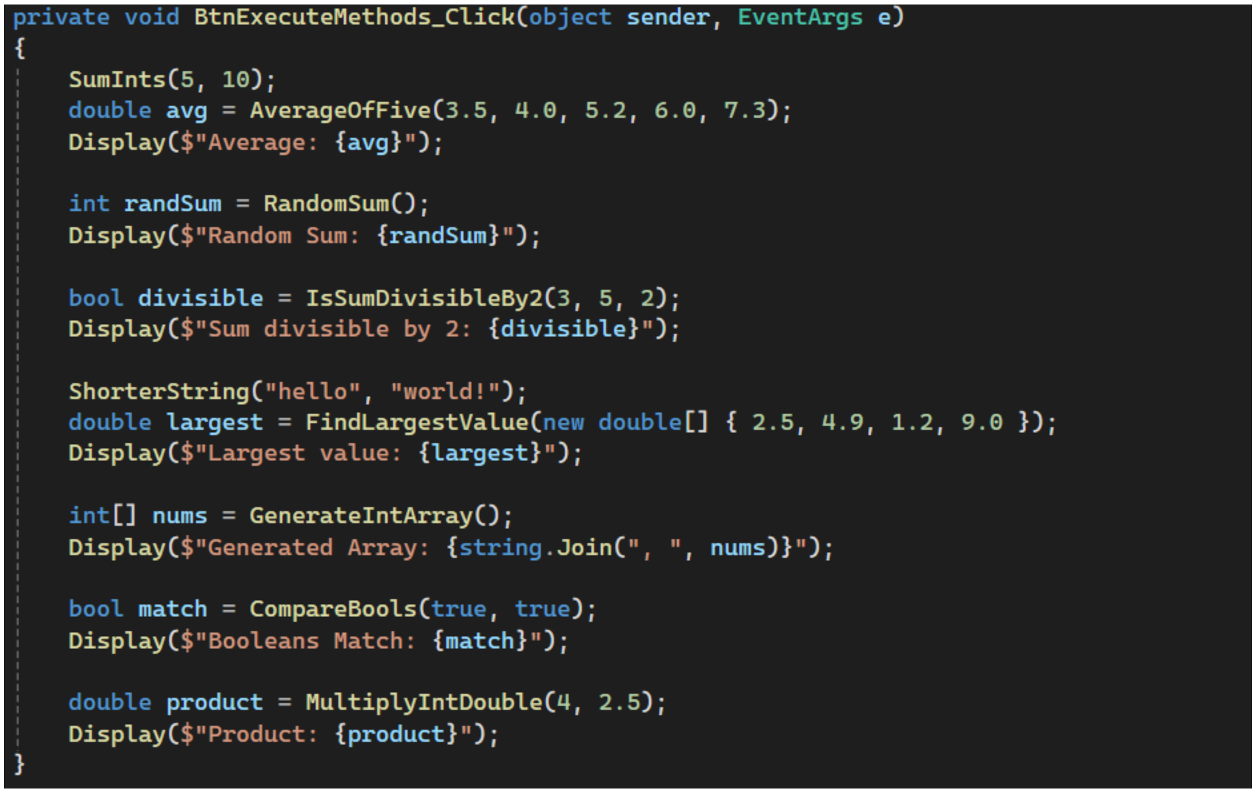


```
Sum of 5 and 10 is 15
Average: 5.2
Random Sum: 162
Sum divisible by 2: True
Shorter string: hello
Largest value: 9
Generated Array: 17, 49, 96, 92, 78, 62, 64, 93, 74, 21
Booleans Match: True
Product: 10
```

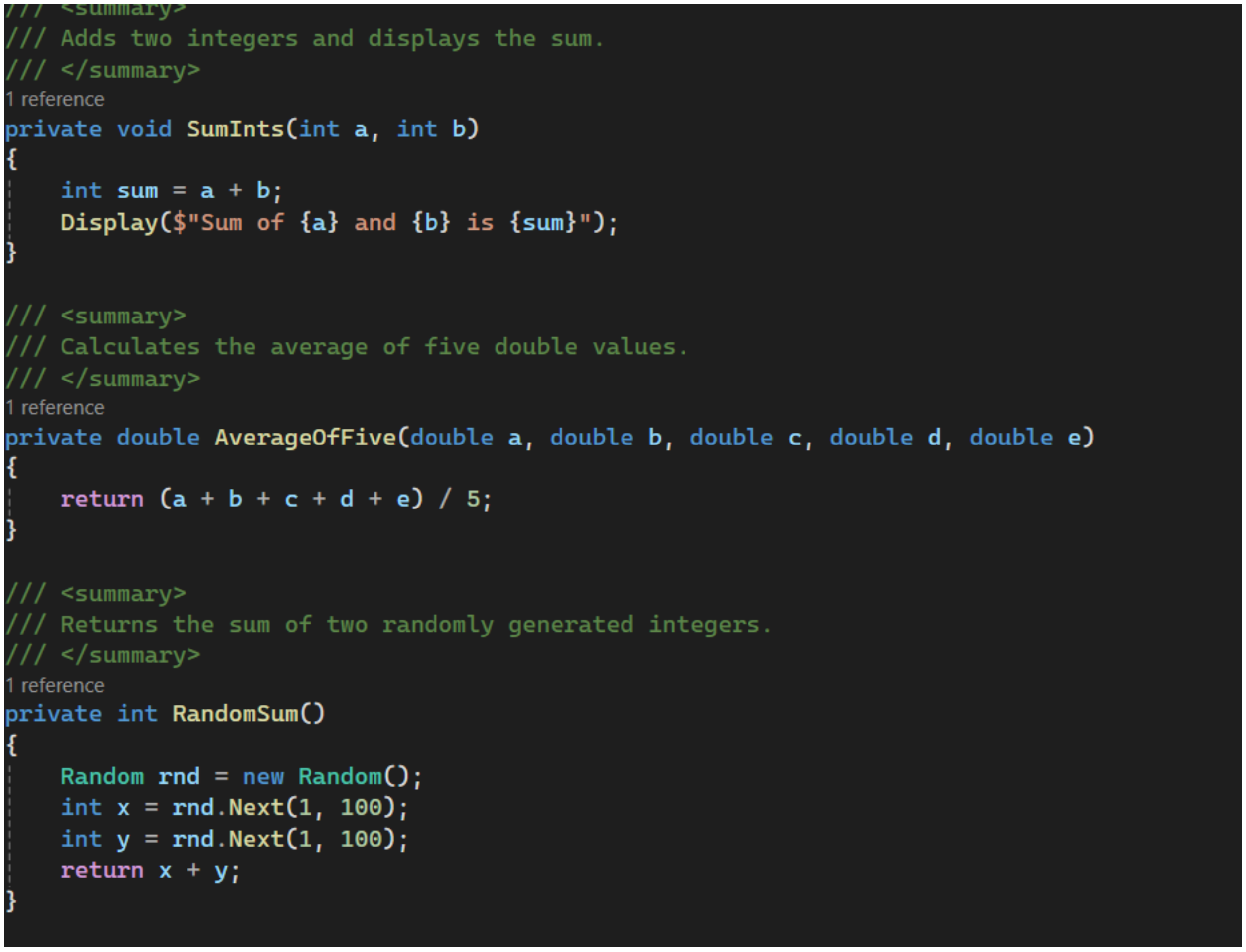
Form after executing all methods.

EXPLANATION;
This screenshot shows how results from each method are appended to the label.
Outputs are formatted and stacked vertically, showing each method’s descriptive result clearly to the user.

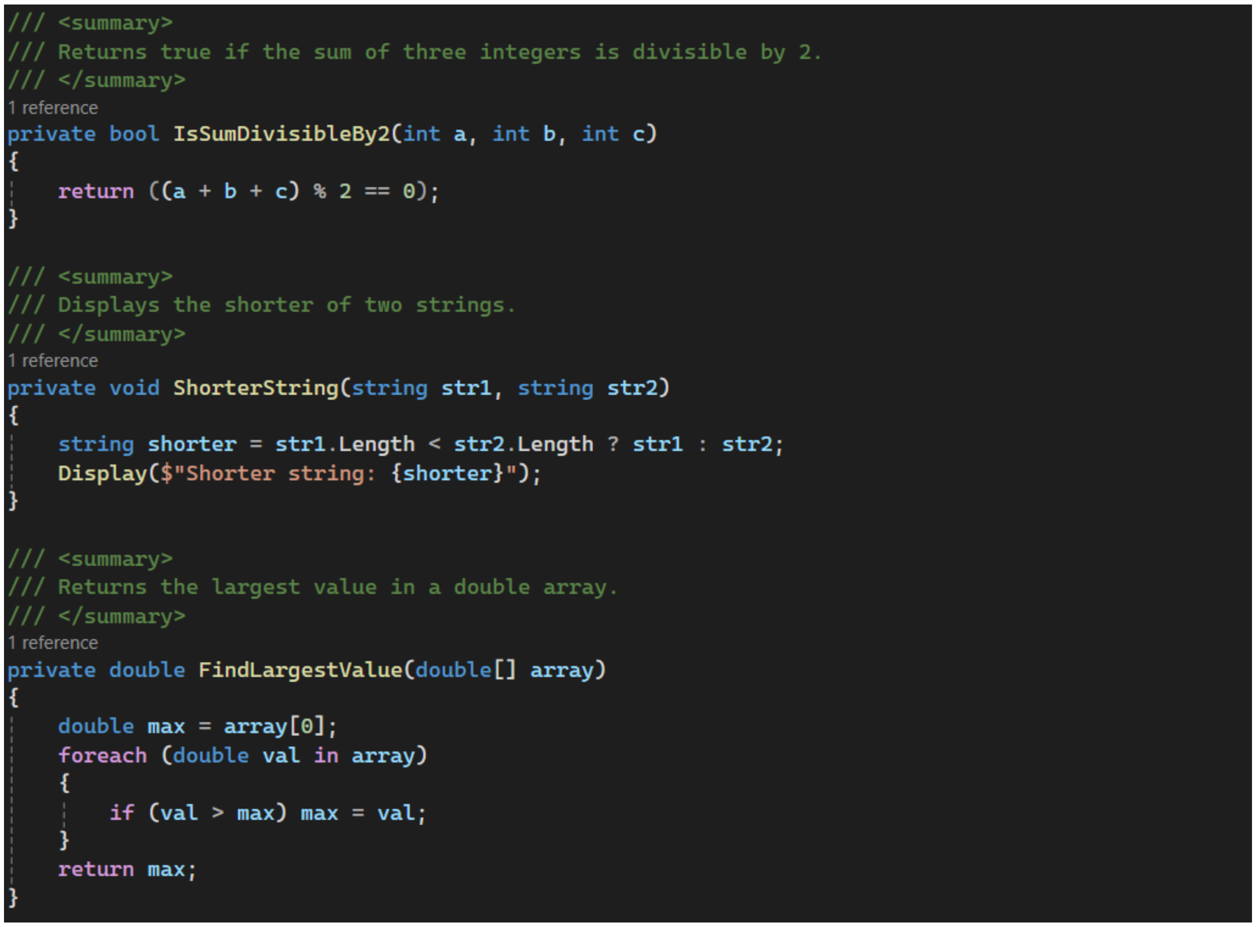
Screenshot(s) of Code Behind



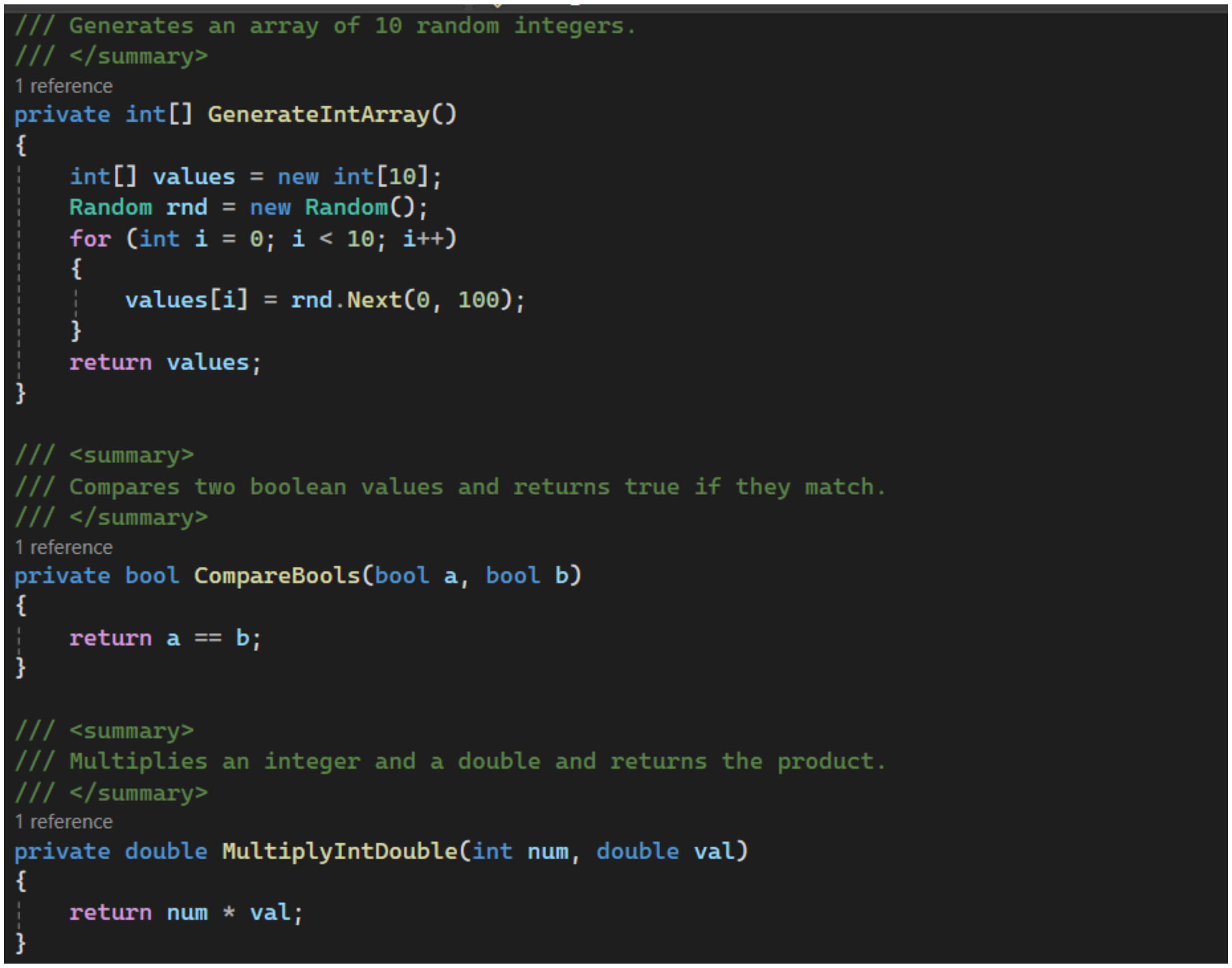
```
private void BtnExecuteMethods_Click(object sender, EventArgs e)
{
    SumInts(5, 10);
    double avg = AverageOfFive(3.5, 4.0, 5.2, 6.0, 7.3);
    Display($"Average: {avg}");

    int randSum = RandomSum();
    Display($"Random Sum: {randSum}");

    bool divisible = IsSumDivisibleBy2(3, 5, 2);
    Display($"Sum divisible by 2: {divisible}");

    ShorterString("hello", "world!");
    double largest = FindLargestValue(new double[] { 2.5, 4.9, 1.2, 9.0 });
    Display($"Largest value: {largest}");

    int[] nums = GenerateIntArray();
    Display($"Generated Array: {string.Join(", ", nums)}");

    bool match = CompareBools(true, true);
    Display($"Booleans Match: {match}");

    double product = MultiplyIntDouble(4, 2.5);
    Display($"Product: {product}");
}
```



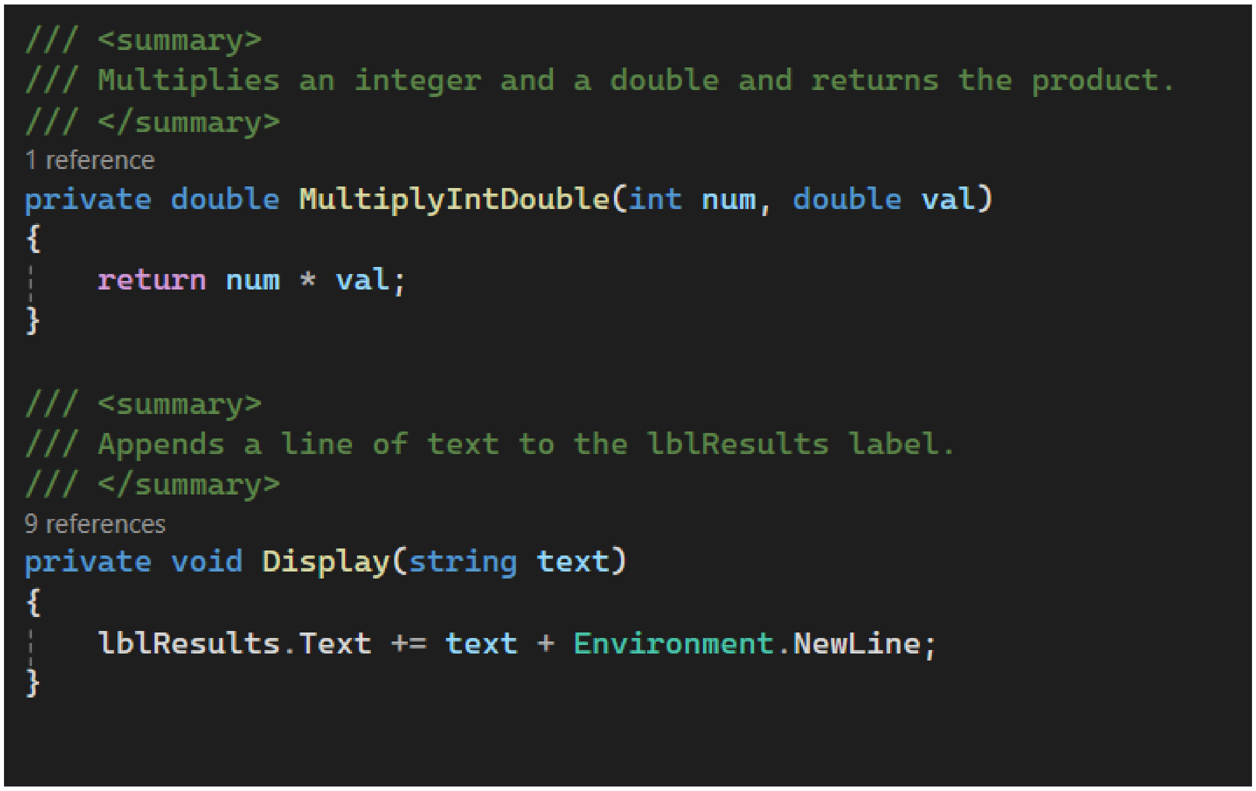
```
/// <summary>
/// Adds two integers and displays the sum.
/// </summary>
1 reference
private void SumInts(int a, int b)
{
    int sum = a + b;
    Display($"Sum of {a} and {b} is {sum}");
}

/// <summary>
/// Calculates the average of five double values.
/// </summary>
1 reference
private double AverageOfFive(double a, double b, double c, double d, double e)
{
    return (a + b + c + d + e) / 5;
}

/// <summary>
/// Returns the sum of two randomly generated integers.
/// </summary>
1 reference
private int RandomSum()
{
    Random rnd = new Random();
    int x = rnd.Next(1, 100);
    int y = rnd.Next(1, 100);
    return x + y;
}
```



```csharp
/// <summary>
/// Returns true if the sum of three integers is divisible by 2.
/// </summary>
1 reference
private bool IsSumDivisibleBy2(int a, int b, int c)
{
    return ((a + b + c) % 2 == 0);
}

/// <summary>
/// Displays the shorter of two strings.
/// </summary>
1 reference
private void ShorterString(string str1, string str2)
{
    string shorter = str1.Length < str2.Length ? str1 : str2;
    Display($"Shorter string: {shorter}");
}

/// <summary>
/// Returns the largest value in a double array.
/// </summary>
1 reference
private double FindLargestValue(double[] array)
{
    double max = array[0];
    foreach (double val in array)
    {
        if (val > max) max = val;
    }
    return max;
}
```



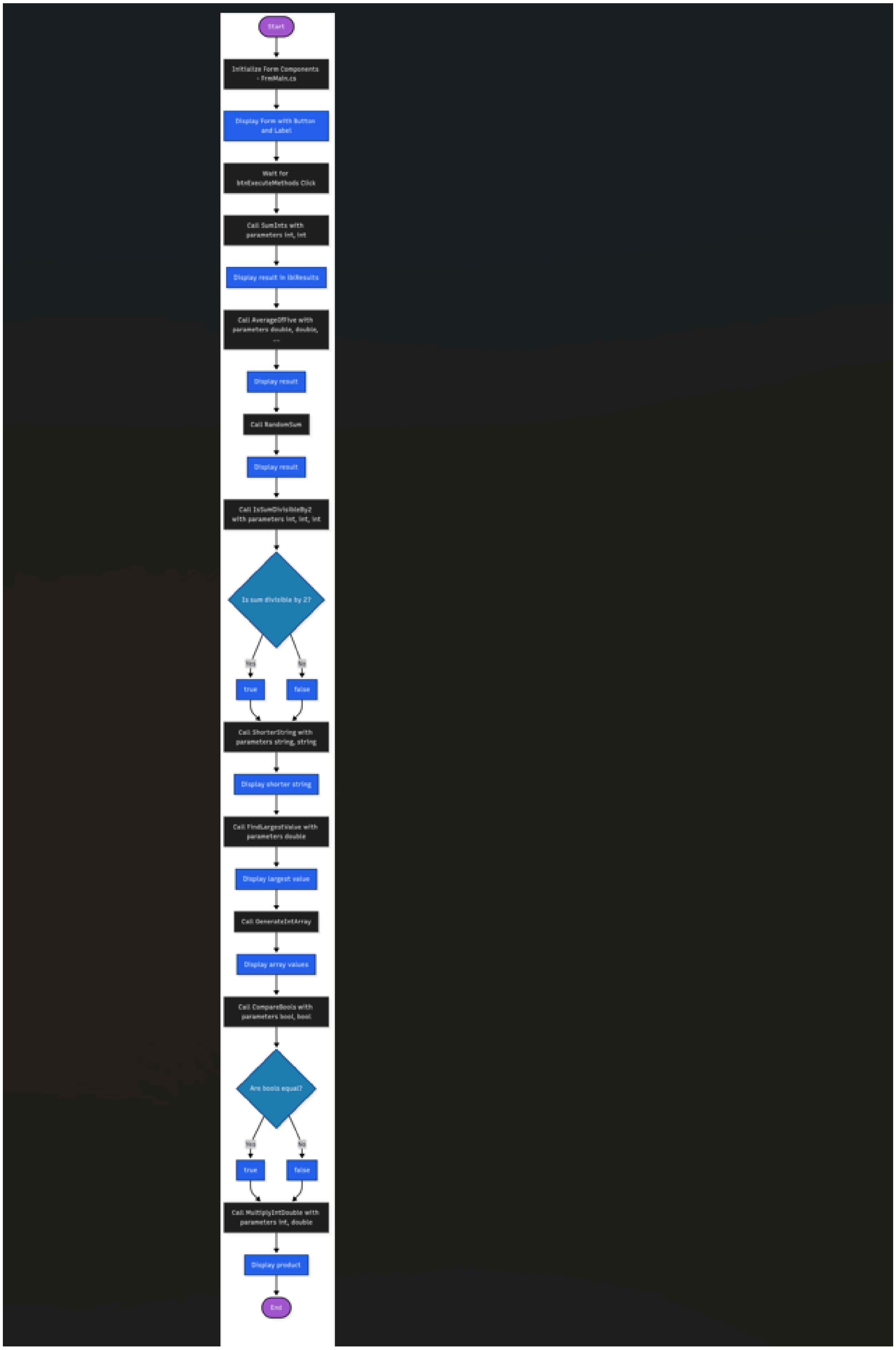
```
/// Generates an array of 10 random integers.
/// </summary>
1 reference
private int[] GenerateIntArray()
{
    int[] values = new int[10];
    Random rnd = new Random();
    for (int i = 0; i < 10; i++)
    {
        values[i] = rnd.Next(0, 100);
    }
    return values;
}

/// <summary>
/// Compares two boolean values and returns true if they match.
/// </summary>
1 reference
private bool CompareBools(bool a, bool b)
{
    return a == b;
}

/// <summary>
/// Multiplies an integer and a double and returns the product.
/// </summary>
1 reference
private double MultiplyIntDouble(int num, double val)
{
    return num * val;
}
```



```
/// <summary>
/// Multiplies an integer and a double and returns the product.
/// </summary>
1 reference
private double MultiplyIntDouble(int num, double val)
{
    return num * val;
}

/// <summary>
/// Appends a line of text to the lblResults label.
/// </summary>
9 references
private void Display(string text)
{
    lblResults.Text += text + Environment.NewLine;
}
```

Multiplies an integer and a double and returns the product.
Appends a line of text to the lblResults label.

Part 3.


Start
Initialize Form Components - FrmMain.cs
Display Form with Button and Label
Wait for btnExecuteMethods Click
Call SumInts with parameters int, int
Display result in lblResults
Call AverageOfFive with parameters double, double, ...
Display result
Call RandomSum
Display result
Call IsSumDivisibleBy2 with parameters int, int, int
Is sum divisible by 2?
Yes
No
true
false
Call ShorterString with parameters string, string
Display shorter string
Call FindLargestValue with parameters double
Display largest value
Call GenerateIntArray
Display array values
Call CompareBools with parameters bool, bool
Are bools equal?
Yes
No
true
false
Call MultiplyIntDouble with parameters int, double
Display product
End

Flowchart showing the data flow and method execution in the Part 2 application.

EXPLANATION:

This flowchart maps the logical sequence of the Windows Forms application.

It begins with a button event and proceeds through a series of method calls.

Each method demonstrates input/output flow using parameters, return values, decisions, and results displayed to the user.

Conditional checks like boolean comparisons and divisibility tests are shown as decision nodes, ensuring clarity in logic.