Activity 4 Part 2

Ian M. McConihay

College of Science, Engineering and Technology, Grand Canyon University

CST-150: C# Programming I

Mark Smithers

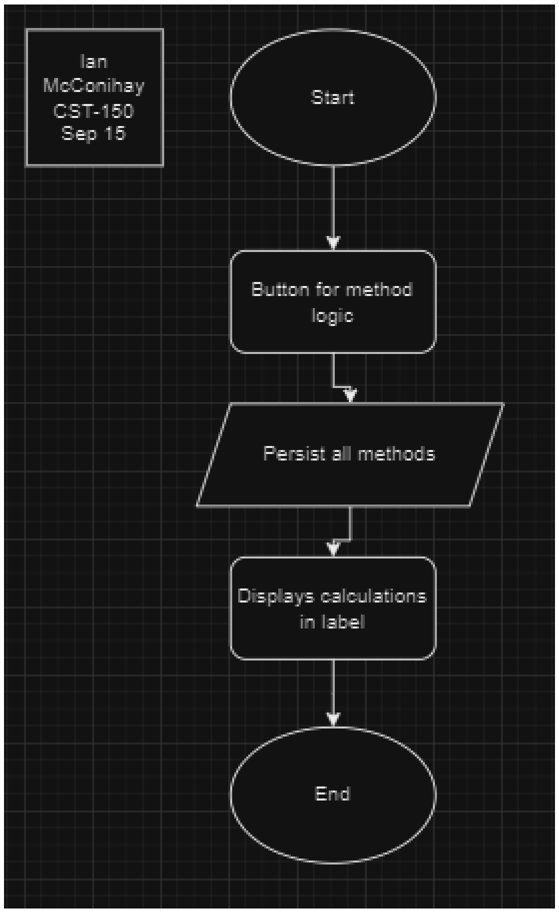
September 15, 2024

Video Link:

https://www.loom.com/share/9b0b7c3f7308468baf9aae110c6f1905?sid=dcf52a5b-64a7-4554-aa3a-c4b0ab24ff90

Github: https://github.com/Ian-McConihay/CST-150

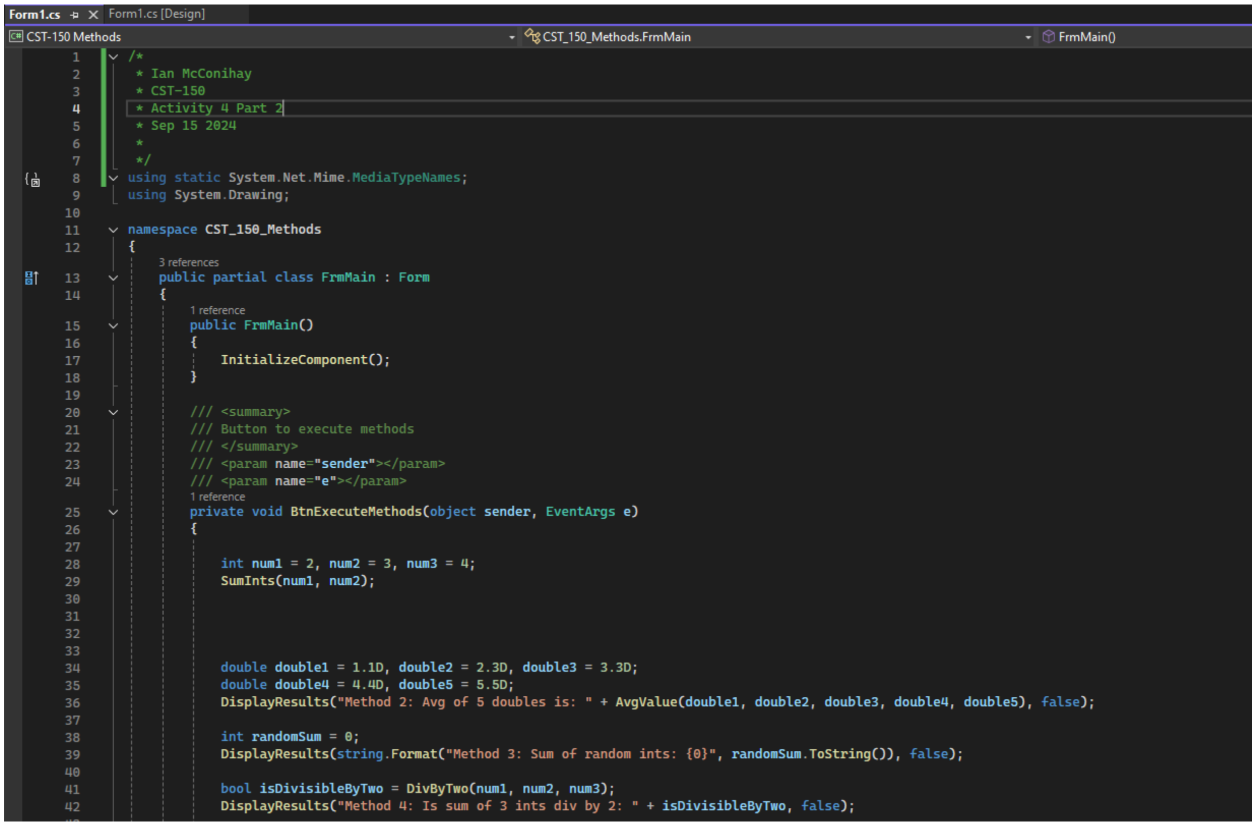
Flowchart



The flow chart for the Activity 4 Part 2 application. This application allows the user to click a button to persist some methods. Once the button is clicked, the user can select a file, and an event fires off to display the text information.

Application Screenshots

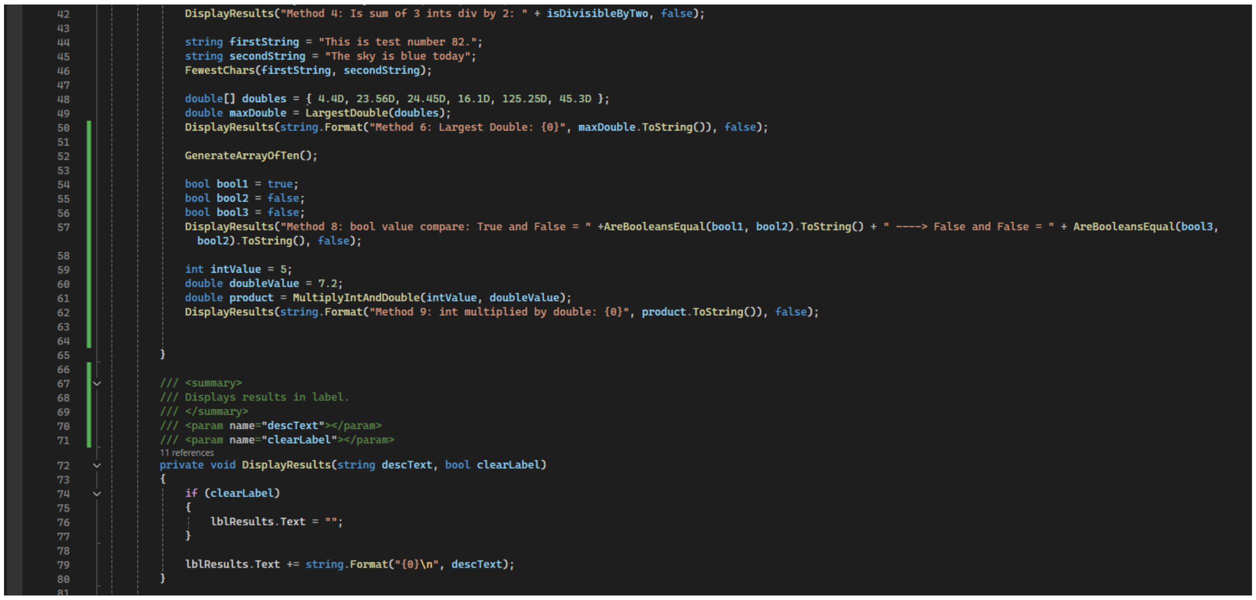
Figure 1: Code



```
/*
 * Ian McConihay
 * CST-150
 * Activity 4 Part 2
 * Sep 15 2024
 *
 */
using static System.Net.Mime.MediaTypeNames;
using System.Drawing;

namespace CST_150_Methods
{
    3 references
    public partial class FrmMain : Form
    {
        1 reference
        public FrmMain()
        {
            InitializeComponent();
        }

        /// <summary>
        /// Button to execute methods
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        1 reference
        private void BtnExecuteMethods(object sender, EventArgs e)
        {

            int num1 = 2, num2 = 3, num3 = 4;
            SumInts(num1, num2);




            double double1 = 1.1D, double2 = 2.3D, double3 = 3.3D;
            double double4 = 4.4D, double5 = 5.5D;
            DisplayResults("Method 2: Avg of 5 doubles is: " + AvgValue(double1, double2, double3, double4, double5), false);

            int randomSum = 0;
            DisplayResults(string.Format("Method 3: Sum of random ints: {0}", randomSum.ToString()), false);

            bool isDivisibleByTwo = DivByTwo(num1, num2, num3);
            DisplayResults("Method 4: Is sum of 3 ints div by 2: " + isDivisibleByTwo, false);
```

In this screenshot we can see the citation. After that we have the main button method. This event fires off all of the other methods and all so has a lot of set variables.

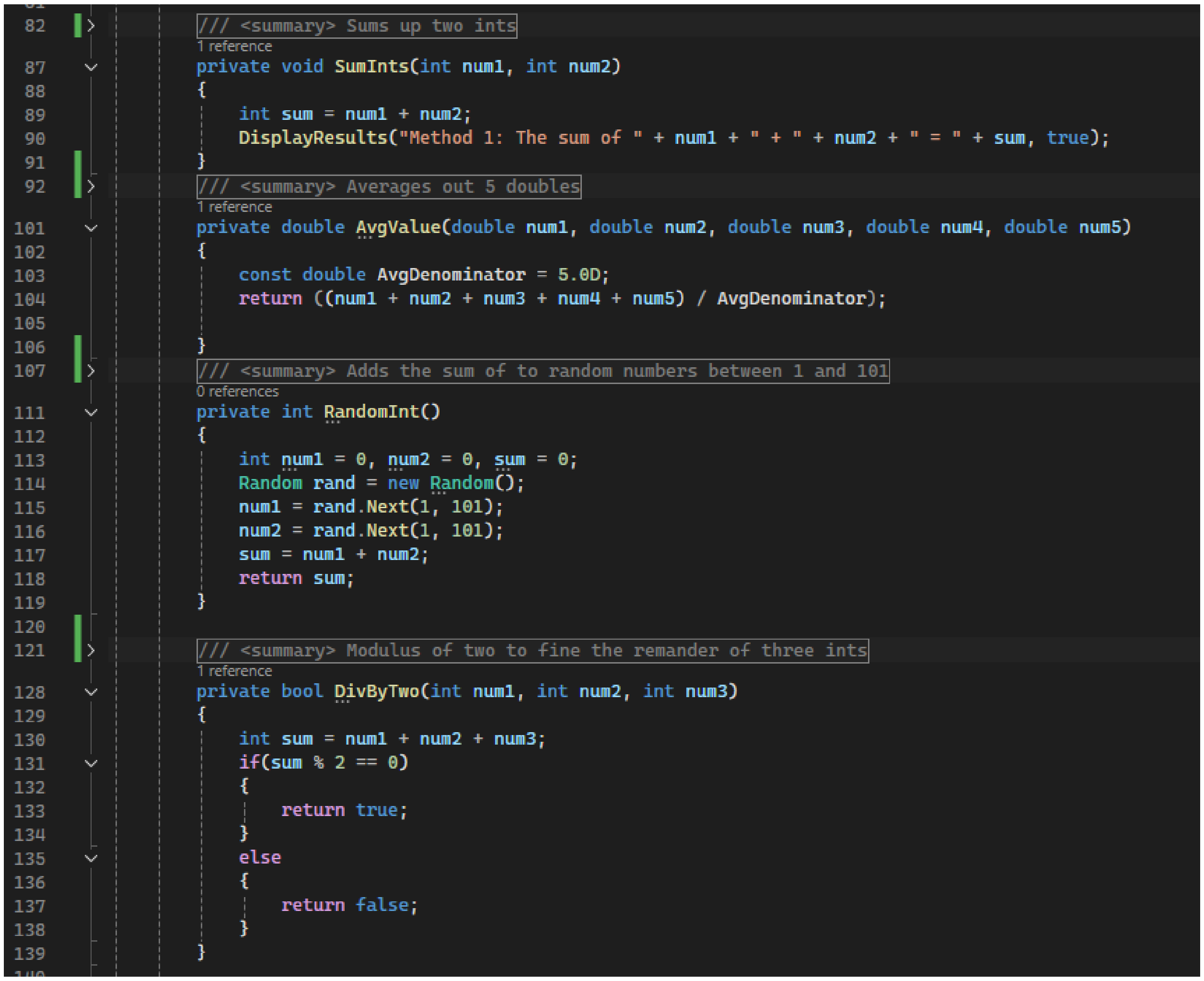
Figure 2: Code



```
        DisplayResults("Method 4: Is sum of 3 ints div by 2: " + isDivisibleByTwo, false);

        string firstString = "This is test number 82.";
        string secondString = "The sky is blue today";
        FewestChars(firstString, secondString);

        double[] doubles = { 4.4D, 23.56D, 24.45D, 16.1D, 125.25D, 45.3D };
        double maxDouble = LargestDouble(doubles);
        DisplayResults(string.Format("Method 6: Largest Double: {0}", maxDouble.ToString()), false);

        GenerateArrayOfTen();

        bool bool1 = true;
        bool bool2 = false;
        bool bool3 = false;
        DisplayResults("Method 8: bool value compare: True and False = " +AreBooleansEqual(bool1, bool2).ToString() + " ----> False and False = " + AreBooleansEqual(bool3,
          bool2).ToString(), false);

        int intValue = 5;
        double doubleValue = 7.2;
        double product = MultiplyIntAndDouble(intValue, doubleValue);
        DisplayResults(string.Format("Method 9: int multiplied by double: {0}", product.ToString()), false);


    }

    /// <summary>
    /// Displays results in label.
    /// </summary>
    /// <param name="descText"></param>
    /// <param name="clearLabel"></param>
    11 references
    private void DisplayResults(string descText, bool clearLabel)
    {
        if (clearLabel)
        {
            lblResults.Text = "";
        }

        lblResults.Text += string.Format("{0}\n", descText);
    }
```

In this screenshot we have the continuation of the event. We also have our display results method that all the methods use. This persists the string in a lblResults label.

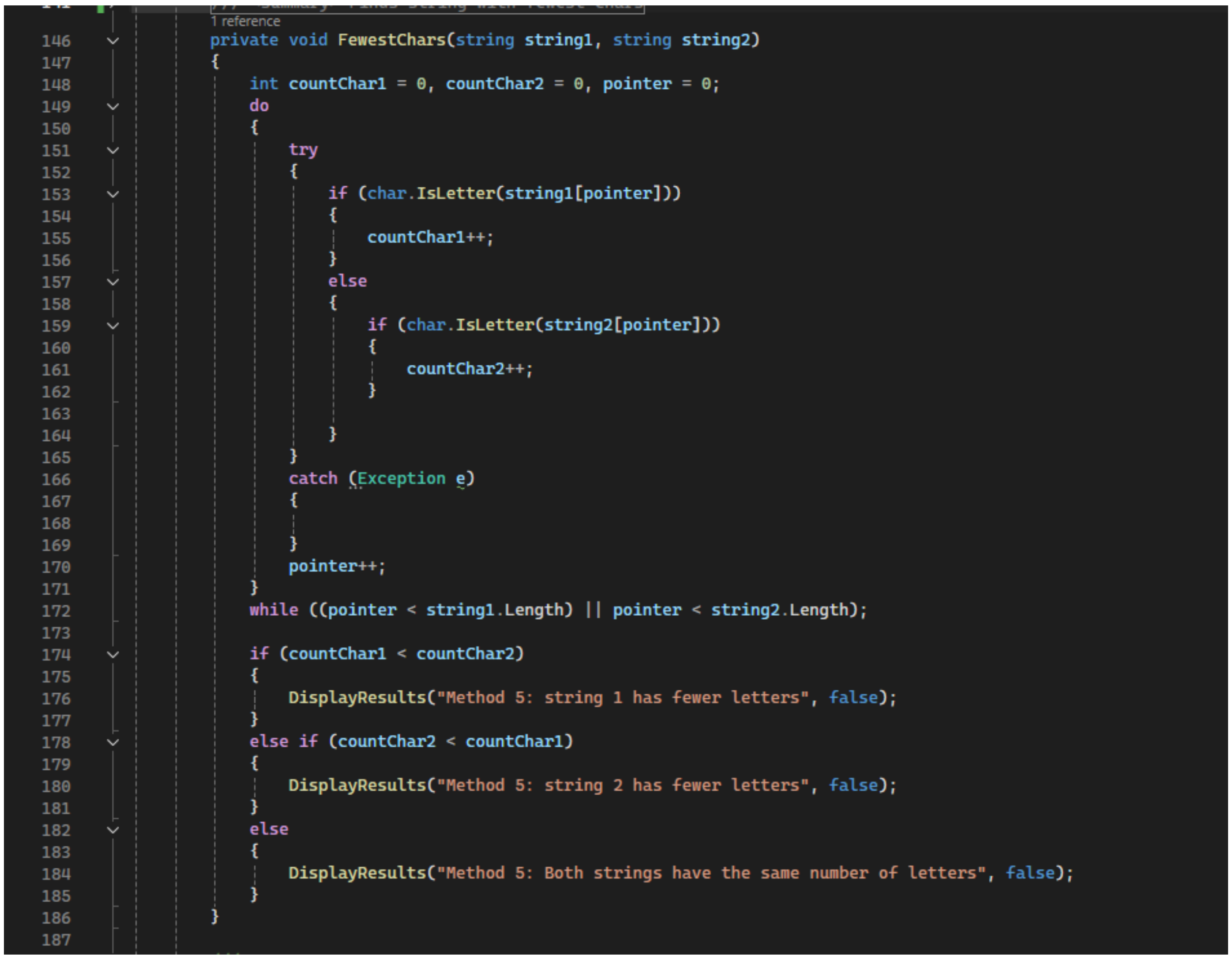
Figure 3: Code



```
/// <summary> Sums up two ints
1 reference
private void SumInts(int num1, int num2)
{
    int sum = num1 + num2;
    DisplayResults("Method 1: The sum of " + num1 + " + " + num2 + " = " + sum, true);
}
/// <summary> Averages out 5 doubles
1 reference
private double AvgValue(double num1, double num2, double num3, double num4, double num5)
{
    const double AvgDenominator = 5.0D;
    return ((num1 + num2 + num3 + num4 + num5) / AvgDenominator);

}
/// <summary> Adds the sum of to random numbers between 1 and 101
0 references
private int RandomInt()
{
    int num1 = 0, num2 = 0, sum = 0;
    Random rand = new Random();
    num1 = rand.Next(1, 101);
    num2 = rand.Next(1, 101);
    sum = num1 + num2;
    return sum;
}

/// <summary> Modulus of two to fine the remander of three ints
1 reference
private bool DivByTwo(int num1, int num2, int num3)
{
    int sum = num1 + num2 + num3;
    if(sum % 2 == 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

SumInts Takes in two integers and adds them together. AvgValue takes in five doubles and finds the average. Next, we have RandomInt that takes two random 1 through 101 numbs and adds them together.

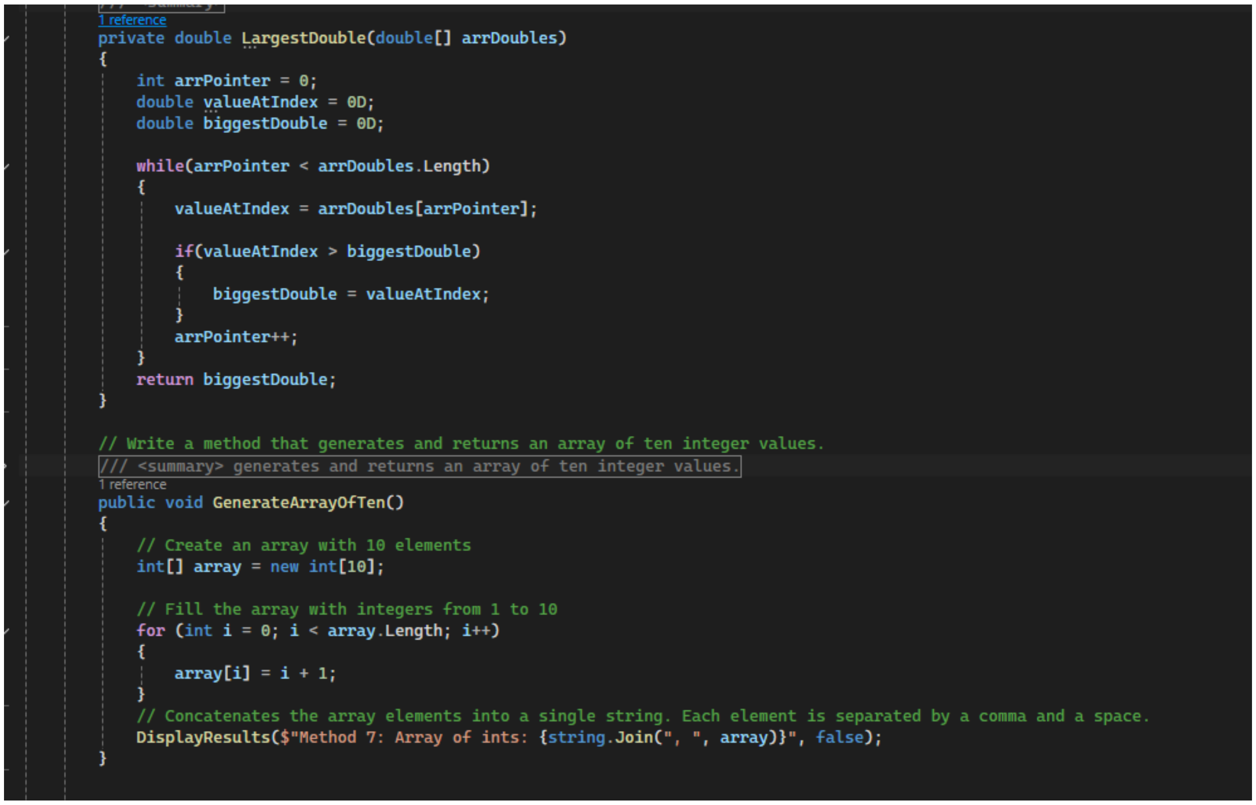
Figure 4: Code



```
1 reference
private void FewestChars(string string1, string string2)
{
    int countChar1 = 0, countChar2 = 0, pointer = 0;
    do
    {
        try
        {
            if (char.IsLetter(string1[pointer]))
            {
                countChar1++;
            }
            else
            {
                if (char.IsLetter(string2[pointer]))
                {
                    countChar2++;
                }

            }
        }
        catch (Exception e)
        {

        }
        pointer++;
    }
    while ((pointer < string1.Length) || pointer < string2.Length);

    if (countChar1 < countChar2)
    {
        DisplayResults("Method 5: string 1 has fewer letters", false);
    }
    else if (countChar2 < countChar1)
    {
        DisplayResults("Method 5: string 2 has fewer letters", false);
    }
    else
    {
        DisplayResults("Method 5: Both strings have the same number of letters", false);
    }
}
```

FewestChars takes into strings. This first gives the string values. Then iterates though them to display different string results.

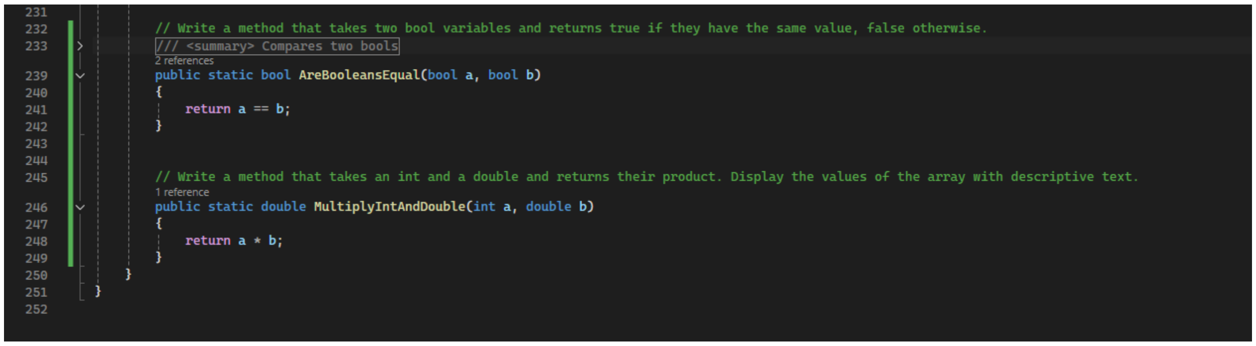
Figure 5: Code



```
1 reference
private double LargestDouble(double[] arrDoubles)
{
    int arrPointer = 0;
    double valueAtIndex = 0D;
    double biggestDouble = 0D;

    while(arrPointer < arrDoubles.Length)
    {
        valueAtIndex = arrDoubles[arrPointer];

        if(valueAtIndex > biggestDouble)
        {
            biggestDouble = valueAtIndex;
        }
        arrPointer++;
    }
    return biggestDouble;
}

// Write a method that generates and returns an array of ten integer values.
/// <summary> generates and returns an array of ten integer values.
1 reference
public void GenerateArrayOfTen()
{
    // Create an array with 10 elements
    int[] array = new int[10];

    // Fill the array with integers from 1 to 10
    for (int i = 0; i < array.Length; i++)
    {
        array[i] = i + 1;
    }
    // Concatenates the array elements into a single string. Each element is separated by a comma and a space.
    DisplayResults($"Method 7: Array of ints: {string.Join(", ", array)}", false);
}
```

Largest double stakes in a double array. Then it iterates through the array and finds the largest double. The display happens in the button method. Then we have a method that generates an array of ten ints.

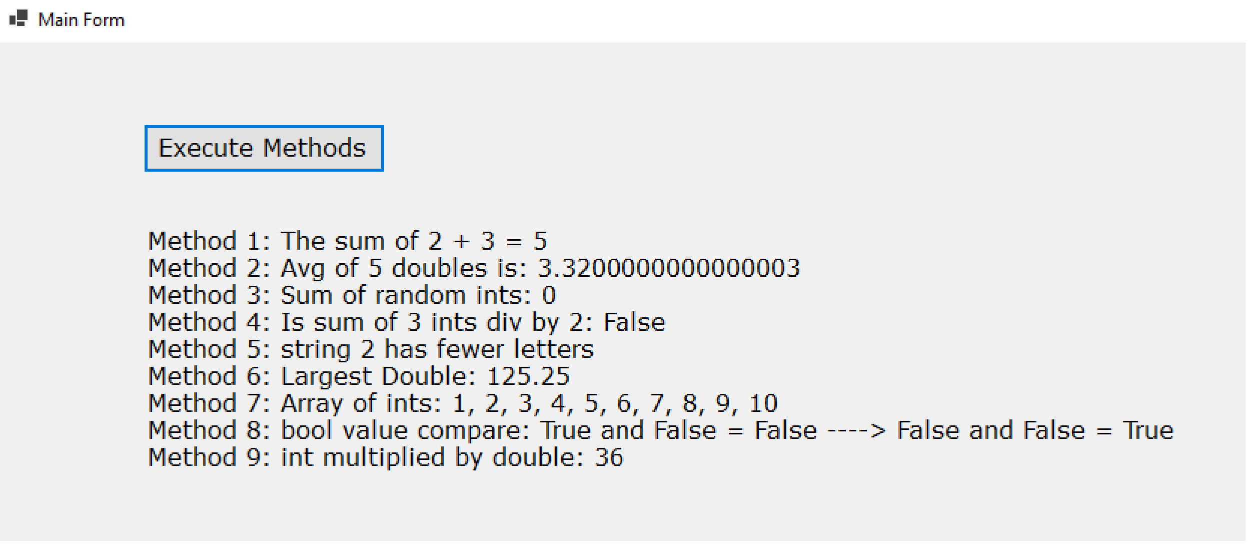
Figure 6: Code



```
        // Write a method that takes two bool variables and returns true if they have the same value, false otherwise.
        /// <summary> Compares two bools
        2 references
        public static bool AreBooleansEqual(bool a, bool b)
        {
            return a == b;
        }


        // Write a method that takes an int and a double and returns their product. Display the values of the array with descriptive text.
        1 reference
        public static double MultiplyIntAndDouble(int a, double b)
        {
            return a * b;
        }
    }
}
```

Lastly, we have AreBooleansEqual. This tasks in two Booleans values and returns if the same value. The last method multiples an int and doubles.

Figure 7: Application Running

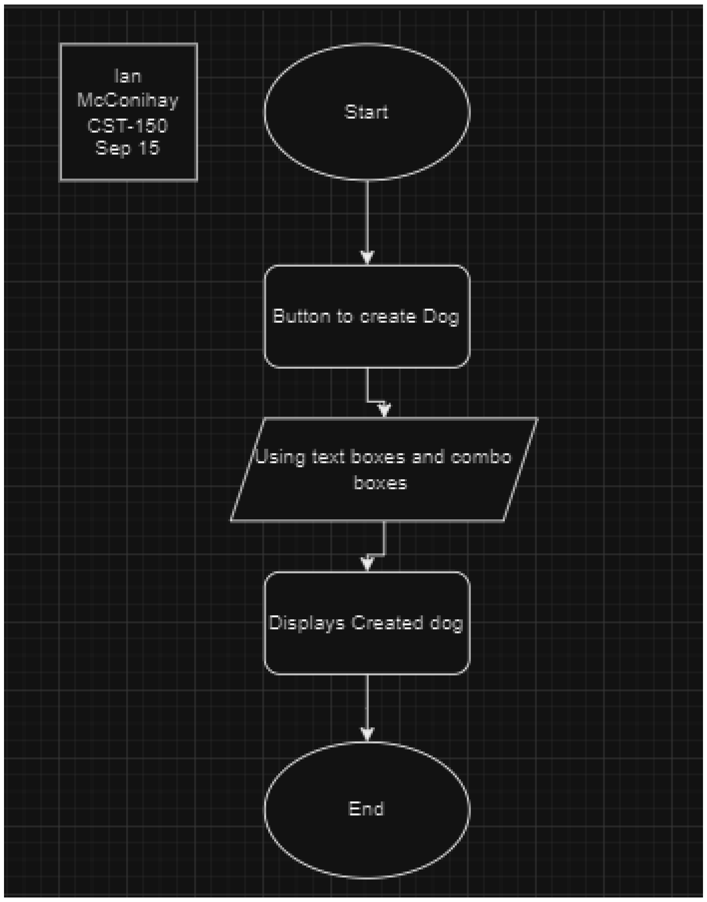


```
Method 1: The sum of 2 + 3 = 5
Method 2: Avg of 5 doubles is: 3.3200000000000003
Method 3: Sum of random ints: 0
Method 4: Is sum of 3 ints div by 2: False
Method 5: string 2 has fewer letters
Method 6: Largest Double: 125.25
Method 7: Array of ints: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
Method 8: bool value compare: True and False = False ----> False and False = True
Method 9: int multiplied by double: 36
```

The application running shows the button that sets off all the methods. All the methods 1 through 9 are shown. I have shown a few different values and the arrays are in a single line.

Part 2 of Activity 4

Flowchart


Ian McConihay CST-150 Sep 15
Start
Button to create Dog
Using text boxes and combo boxes
Displays Created dog
End

Activity 4 part 2 required a flowchart for Activity 5. This application allows the user to create a Dog using a series of components. We also need to make sure the class has the appropriate layers of logic.

What was challenging?

DisplayResults gave me some difficulty because I had not realized I set one method to true. So, the rest of the previous methods had disappeared.

What did you learn?
I learned about method structure.

How would you improve on the project?
I would give text boxes to all the methods to give the user the ability to change the outcome.

How can you use what you learned on the job?
Methods used for calculations are commonly used to save repeated logic.