```
1  //Programmer: Aaron Yoon
2  //Date: 10/12/22
3  //Project: Triangle Checker
4
5  namespace Triangle_Calculator
6  {
7      public partial class Form1 : Form
8      {
9          int trianglesTracked;
10
11         int sideTracked;
12         int scaleneTracked;
13         int equilateralTracked;
14         int isoscelesTracked;
15
16         int angleTracked;
17         int rightTracked;
18         int obtuseTracked;
19         int acuteTracked;
20         public Form1()
21         {
22             InitializeComponent();
23         }
24
25         private void Form1_Load(object sender, EventArgs e)
26         {
27
28         }
29
30         private void btnQuit_Click(object sender, EventArgs e)
31         {
32             this.Close();
33         }
34
35         private void btnCalculate_Click(object sender, EventArgs e)
36         {
37             //declare your variables to make your life easier
38             int length1 = int.Parse(txtLength1.Text);
39             int length2 = int.Parse(txtLength2.Text);
40             int length3 = int.Parse(txtLength3.Text);
41             bool isTriangle;
42
43
44             //make a list for your lengths to use Max and Min methods
45             List<int> triLengths = new List<int>()
46             {
47                 length1,
48                 length2,
49                 length3
```

```
50              };
51
52              //this assigns the sides values in in order of greatest to
                  least
53              int hypotenuse = triLengths.Max();
54              int legBase1 = triLengths.Min();
55              int legBase2 = ((int)(triLengths.Average() * 3 - legBase1 -
                  hypotenuse));
56
57              //output your text
58              lblOutput.Text =
59                  "Hypotenuse: " + hypotenuse.ToString() + "\n"
60                  + "Base: " + legBase2.ToString() + "\n"
61                  + "Leg: " + legBase1.ToString() + "\n"
62                  + getTriangle(legBase1, legBase2, hypotenuse) + "\n"
63                  + trianglesTracked.ToString() + " triangles tracked" +
                    "\n"
64                  + rightTracked.ToString() + " right triangles" + "\n"
65                  + (rightTracked * 100 / angleTracked).ToString() + " right
                     percentage" + "\n"
66                  + obtuseTracked.ToString() + " obtuse triangles" + "\n"
67                  + ( obtuseTracked * 100 / angleTracked).ToString() + "
                    obtuse percentage" + "\n"
68                  + acuteTracked.ToString() + " acute triangles" + "\n"
69                  + (acuteTracked * 100 / angleTracked).ToString() + " acute
                     percentage" + "\n"
70                  + scaleneTracked.ToString() + " scalene triangles" + "\n"
71                  + (scaleneTracked * 100 / sideTracked).ToString() + "
                    scalene percentage" + "\n"
72                  + isoscelesTracked.ToString() + " isosceles triangles" +
                    "\n"
73                  + (isoscelesTracked* 100 / sideTracked).ToString() + "
                    iscosceles percentage" + "\n"
74                  + equilateralTracked.ToString() + " equilateral triangles"
                     + "\n"
75                  + (equilateralTracked * 100 / sideTracked).ToString() + "
                    equilateral percentage";
76
77
78
79          }
80
81          public string getTriangle(int leg1, int leg2, int hyp)
82          {
83              //this methods gets what type of triangle it is or if it even
                  is a triangle
84              //it takes all sides as its args
85
86              //string for what angle type it is
```

```
87              string angle;
88              //string for what side type it is
89              string side;
90              //bool for if it is a triangle
91              bool isTriangle = false;
92
93              //these if statements check what angle type it is
94              if (Math.Pow(leg1, 2) + Math.Pow(leg2, 2) == Math.Pow(hyp, 2))
95              {
96                  angleTracked += 1;
97                  rightTracked += 1;
98                  angle = "Right Triangle";
99                  isTriangle = true;
100                 trianglesTracked += 1;
101             }
102
103             else if (Math.Pow(leg1, 2) + Math.Pow(leg2, 2) < Math.Pow(hyp, ↵
                    2) && leg1 + leg2 > hyp)
104             {
105                 angleTracked += 1;
106                 obtuseTracked += 1;
107                 angle = "Obtuse Triangle";
108                 isTriangle = true;
109                 trianglesTracked += 1;
110             }
111
112             else if (Math.Pow(leg1, 2) + Math.Pow(leg2, 2) > Math.Pow(hyp, ↵
                    2) && leg1 + leg2 > hyp)
113             {
114                 angleTracked += 1;
115                 acuteTracked += 1;
116                 angle = "Acute Triangle";
117                 isTriangle = true;
118                 trianglesTracked += 1;
119             }
120
121             else if (leg1 + leg2 <= hyp)
122             {
123                 angle = "your triple cannot make a triangle!";
124                 isTriangle = false;
125                 MessageBox.Show("Hypotenuse is greater than both legs      ↵
                      combined!",
126                   "ERROR",
127                   MessageBoxButtons.OK,
128                   MessageBoxIcon.Error
129                   );
130             }
131             else
132                 angle = null;
```

```
133
134             //these if statements checks what side type it is
135             if (leg1 == leg2 && leg1 != hyp && isTriangle)
136             {
137                 sideTracked += 1;
138                 isoscelesTracked += 1;
139                 side = "Isosceles ";
140             }
141             else if (leg1 == leg2 && leg1 == hyp && isTriangle)
142             {
143                 sideTracked += 1;
144                 equilateralTracked += 1;
145                 side = "Equilateral ";
146             }
147             else if (isTriangle)
148             {
149                 sideTracked += 1;
150                 scaleneTracked += 1;
151                 side = "Scalene ";
152             }
153             else
154                 side = "ERROR: ";
155             //gives back what triangle it is
156             return side + angle;
157         }
158
159         /*public string getSideTriangle(int leg1, int leg2, int hyp)
160         {
161             //this method returns what side type your triangle is
162             if (leg1 == leg2 && leg1 != hyp)
163                 return "Isosceles ";
164             else if (leg1 == leg2 && leg1 == hyp)
165                 return "Equilateral ";
166             else
167                 return "Scalene ";
168         }
169         */
170     }
171 }
```