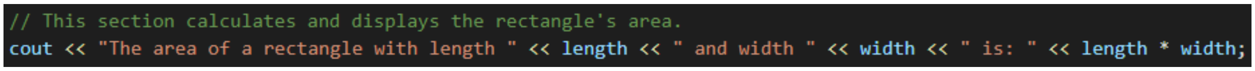
Describe how functions helped to make your program more organized. Discuss any challenges you faced when passing parameters and how you resolved them. Be specific about your thought process and problem-solving strategies. Come up with one additional feature or variation to the program. Describe this feature and why you added it.

Toby Hansen - April 4, 2025
Module 9 Individual Assignment Reflection

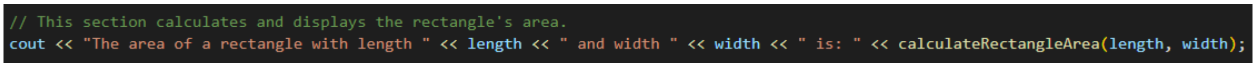
I think the idea of this assignment was to show how functions can help keep code more clean and organized, However it is my opinion that this wasn't a good project to make that point with. This project is all about calculating the areas of various shapes, and each shape has a function that takes parameters for sides of the shape and returns the area of said shape. This sounds reasonable at first, but each function is only called a single time throughout the entire code! What this means is that the functions are completely unnecessary and only serve to add more clutter to the program; they do *not* make the program more organized.

Adding these specific functions means that instead of writing all our code in a single pretty line:

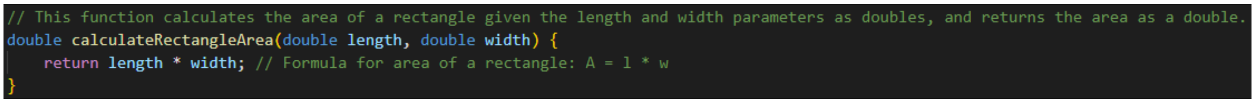


```
// This section calculates and displays the rectangle's area.
cout << "The area of a rectangle with length " << length << " and width " << width << " is: " << length * width;
```

We had to add a random call to perform the calculation elsewhere:



```
// This section calculates and displays the rectangle's area.
cout << "The area of a rectangle with length " << length << " and width " << width << " is: " << calculateRectangleArea(length, width);
```



```
// This function calculates the area of a rectangle given the length and width parameters as doubles, and returns the area as a double.
double calculateRectangleArea(double length, double width) {
    return length * width; // Formula for area of a rectangle: A = l * w
}
```

I'm not saying that the former solution is always good practice, but for an extremely small project like this, writing calculateArea functions is redundant. Functions are good for reducing repetitiveness, but there are no repeated calls to any of the functions.

I faced no challenges with passing parameters; the syntax is identical to C# which I'm fluent in.

I added two custom features. I added the ability to calculate the area of a trapezoid, and to do another calculation after finishing the program. I chose the trapezoid as another shape because the formula for its area `(A = (s1 + s2) * h / 2)` is the next simplest after rectangle triangle and circle. I was thinking of doing a pentagon area calculator, but figured it was too complicated `(A = sqrt(5 * (2 * sqrt(5) + 5)) * s^2 / 4)` compared to the initial three shapes. I chose to add the repetition feature so that the user has the option of doing another calculation without having to close and reopen the console window. This was especially helpful while QA testing to ensure that all edge cases were handled.