NET3010 Lab 5: JavaScript Coding Challenge

Note you will need to complete this lab by uploading all required files onto BrightSpace. You are responsible for the content of these lab instructions on your weekly quizzes and exams.

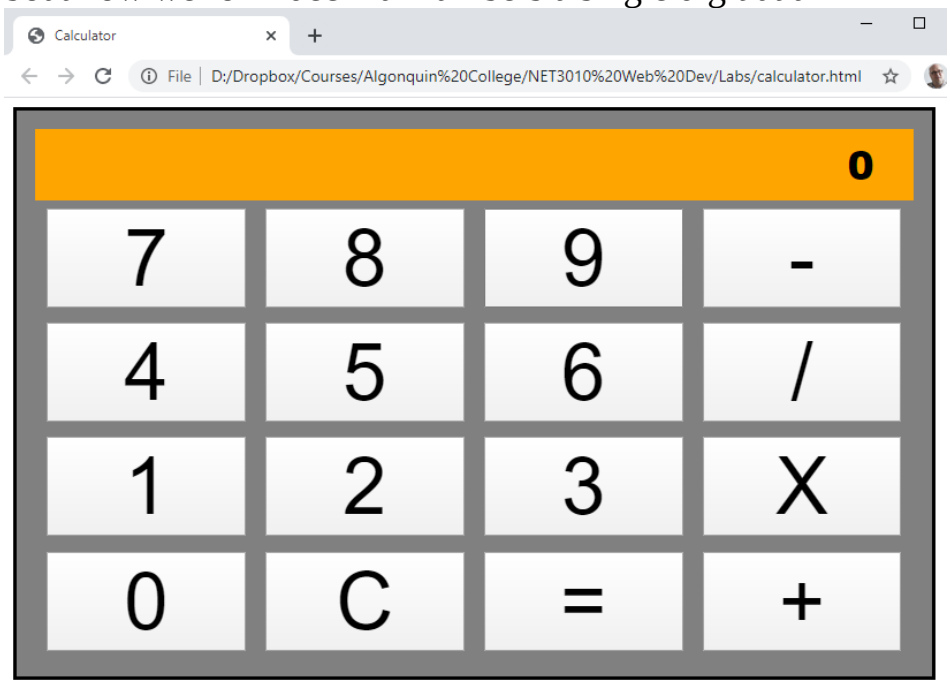
Read the entire lab instruction before starting.

This lab is to be completed on BrightSpace any lab worksheets handed in will be discarded. Carefully follow the procedures outlined in this lab worksheet. **If at any time you are unsure or are having problems, consult your lab instructor**.

Required Equipment: VSCode

Part I: Making a Calculator

In this lab we are going to learn a few JavaScript skills important for web design. I am including in this lab starter html code for a calculator. It is a very basic calculator, but it is intended to work exclusively by you pressing (clicking) buttons. Below is an image of the calculator with a bit of CSS to make it look not so terrible. For this lab, a fully implemented calculator will behave exactly like a cheap store-bought calculator. Note there is to be **no input** other than the buttons – so you will need to think about how we form decimal numbers a single digit at a time. The starter html gives you **all** the buttons that for which I would like you to program the onclick JavaScript code. The calculator display is simply an element that we target with JavaScript and rewrite with what should appear on the screen of our calculator. **Do not turn the display into an input field you will fail the assignment**.



The starter html is as follows:

```
<!doctype html>
<html>
     <head>
          <!-- your meta data -->
          <style>
               <!-- your embedded CSS -->
          </style>
          <title>Very Simple Calculator</title>
     </head>
     <body>
          <div class="calculator">

               <div class="display">
                    0
               </div>

               <div class="keypad">
                    <button>7</button><button>8</button>
                    <button>9</button><button>-</button><br>
                    <button>4</button><button>5</button>
                    <button>6</button><button>/</button><br>
                    <button>1</button><button>2</button>
                    <button>3</button><button>X</button><br>
                    <button>0</button><button>C</button>
                    <button>=</button><button>+</button>
               </div>

          </div>
     </body>
</html>
```

Please use appropriate CSS formatting to make your calculator look like a calculator with clear buttons and an attractive layout. I achieved the layout on the previous page using less than 20 lines of CSS and by removing the **
** tags which are just there for simple formatting before you make it pretty. You should use CSS layout elements to create a responsive calculator that looks good as a small web app on your page. Also, an appropriate font stack will really help to sell the calculator app aspect of this project. For this exercise you will write all your CSS and your JavaScript in the one html file. (However, please submit a zipped file at the end of the lab as BrightSpace will eat your code otherwise!) Here are the parameters for how the calculator should function.

The display should start with a 0 when it is waiting for an initial value to be entered via the buttons. Pressing the C button (clear) will reset the calculator and should display a 0 in the **`.display`** element (as seen on previous page).

When the user clicks on any of the number keys the displayed value must be updated. If I have previously clicked a 9 then clicking a 3 should change the displayed number to 93.

When a user chooses any of the four operator keys (subtraction (-), division (/), multiplication (X), or addition(+)) then you will complete any previously started operation and display the result and your code will remember that the next number selected will start a new operand for our calculator. Consider the flow of clicks in the chart to the right. Choosing the = key will complete the calculation, display the final value, and reset the calculator for the next equation **without** deleting the result until a new key is pressed.

Button Clicked	Display
3	3
3	33
+	33
4	4
0	40
+	73
2	2
=	75

In order to complete this exercise, you will need to think about how numbers are formed in the base 10 number system. An elegant answer would use about three different functions and possibly a few global variables. Elegance aside, the goal is to have the calculator working correctly.

Bonus points if you can have the subtraction (-) key do double duty by making a negative number when it is used before entering a number. This is quite tricky to get right because there are two cases where it may be invoked.

With your calculator completed and functional, please zip up all the files (**lab6.zip**) created in this lab including a copy of any images that you may have used to make it pretty.

- **calculator.html**
- **any images** used

Upload the zip file to BrightSpace.

Marking Rubric (/28)

Task	Weight
All web pages are all properly formatted for HTML5	1
No inline or linked CSS	1
Appropriate font stack choice for calculator	1
Calculator has a responsive and clean layout	2
Numbers form properly when buttons pressed	2
Operators work correctly	5
Clear button functions correctly	1
Display is **not** an input field	14
File zipped and correctly named	1
[BONUS] Subtract button also forms negative numbers correctly	2