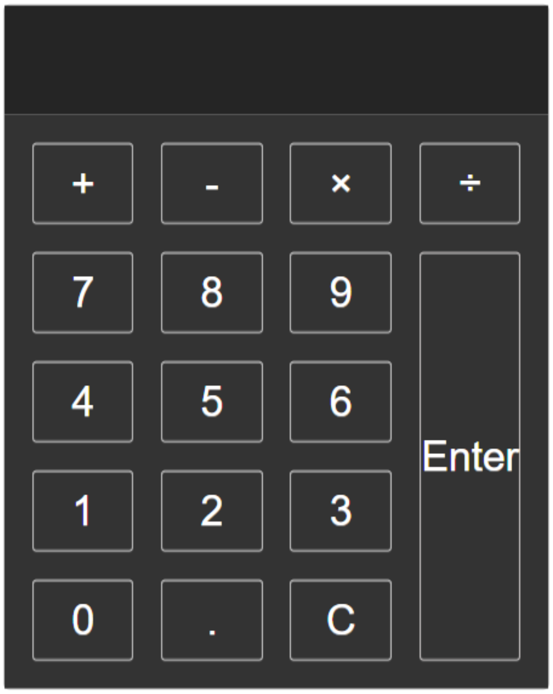
Goal

In this assignment we are going to try to create a reverse polish notation calculator. The main idea is that you allow the user to add numbers onto a "stack". Once you have 2 numbers on the stack, you can start to do operations on the numbers by "popping" them off the stack, doing the operation on them, and returning it to the user.

The calculator should look something like this:



As this lab is not about HTML, the design is totally up to you, I won't be upset if it is not pretty, as the CSS for this style is difficult. The calculator, at a minimum, needs to have:

- A way to enter the numbers
- A way to add a decimal
- The ability to do
    - Addition
    - Subtraction
    - Division
    - Multiplication
- A way to add a number to the stack (Enter)
- A way to clear the current text-box

Bonus (Easier)

If you finish early, try to add some additional functions to the calculator (for example add the modulo operator, which finds the remainder after a division e.g. $5\%2 = 1$).

Bonus (Harder)

If you finish early, try to create a regular calculator using the same HTML/CSS. **Warning:** This is a much more difficult problem than the reverse polish calculator. My advice on how to approach the problem is to allow the user to input data until they press enter, then start at the beginning of the string and do all the calculations. This is a much harder, as you will have to account for the rules of operation (BEDMAS), which requires you to break down the problem into smaller chunks (each individual operation takes 2 values, the left and the right values e.g. $Left + Right = result$) and then rank the operations based on their order in BEDMAS.

For example, $(A/B + A * C)$ must be performed in the order

1. $A/B = v_1$
2. $A * C = v_2$
3. $v_1 + v_2 = result$

To see the visualization for this technique, look at the Calculator_Outline_G2T.pdf file on git.

General Instructions

Create the HTML/CSS

1. Create the calculator HTML and CSS
2. The Calculator should be made up of at least 17 buttons and 1 input field.
3. Assign a class to each type of button and field (e.g. number, decimal, operator, clear, enter)
4. Design the basic layout of the calculator and figure out how you would need to divide up all the tags so it would look proper.
5. Change the coloring of the calculator so it looks good.

Create the JavaScript

1. Create a Calculator object that will change the attributes of the CSS
2. Create the following functions (at a minimum)
    a. **clear** – Clears the screen
    b. **enter** – Adds the input to the stack
    c. **numberHandle** – Handles the input of a number
    d. **operatorHandle** – Handles the input of an operator ($+ - / *$)
    e. **decimalHandle** – Handles the input of a decimal
    f. **calculate** – Main function

Functions You May Need

```
// General Functions and Objects //
document.addEventListener();
document.getElementsByClassName();
target.classList.contains();
console.log();
String()
parseInt()
```

```
// Array Functions & variables //
.push()
.pop()
.length
```