JavaScript Assignments

1. Read more about **JS** and try to answer as many questions as you can from the **JS2 - Terminology & Definitions** file, which you can find attached in the email.

2. **Assignment (Rock Paper Scissors - UI):**
   a) In our UI, the player should be able to play the game by clicking on buttons rather than typing their answer in a prompt.
   b) For now, remove the logic that plays exactly five rounds.
   c) Create three buttons, one for each selection. Add an event listener to the buttons that call your playRound function with the correct playerSelection every time a button is clicked. (you can keep the console.logs for this step)
   d) Add a div for displaying results and change all of your console.logs into DOM methods.
   e) Display the running score, and announce a winner of the game once one player reaches 5 points.

3. **Assignment (Calculator):**
   a) Your calculator is going to contain functions for all of the basic math operators you typically find on simple calculators, so start by creating functions for the following items and testing them in your browser's console.
      a. add
      b. subtract
      c. multiply
      d. divide
   b) Create a new function **operate** that takes an operator and 2 numbers and then calls one of the above functions on the numbers.
   c) Create a basic HTML calculator with buttons for each digit, each of the above functions and an "Equals" key.
      a. There should also be a display for the calculator
      b. Add a "clear" button.
   d) Create the functions that populate the display when you click the number buttons… you should be storing the 'display value' in a variable somewhere for use in the next step.
   e) Make the calculator work! You'll need to store the first number that is input into the calculator when a user presses an operator, and also save which operation has been chosen and then **operate()** on them when the user presses the "=" key.
      a. You should already have the code that can populate the display, so once **operate()** has been called, update the display with the 'solution' to the operation.
      b. Figure out how to store all the values and call the operate function with them.
   f) Gotchas: watch out for and fix these bugs if they show up in your code:

a. Users should be able to string together several operations and get the right answer, with each pair of numbers being evaluated at a time. For example, **12 + 7 - 5 * 3 =** should yield **42**. Here's a good example of how it should look like.
b. Your calculator should not evaluate more than a single pair of numbers at a time. Example: you press a number button (**12**), followed by an operator button (**+**), a second number button (**7**), and finally a second operator button (**-**). Your calculator should then do the following: first, evaluate the first pair of numbers (**12 + 7**), second, display the result of that calculation (**19**), and finally, use that result (**19**) as the first number in your new calculation, along with the next operator (**-**).
c. You should round answers with long decimals so that they don't overflow the screen.
d. Pressing **=** before entering all of the numbers or an operator could cause problems!
e. Pressing "clear" should wipe out any existing data.. make sure the user is really starting fresh after pressing "clear"
f. Display a snarky error message if the user tries to divide by 0… don't let it crash your calculator!

g) Advanced: Users can get floating point numbers if they do the math required to get one, but they can't type them in yet. Add a **.** button and let users input decimals! Make sure you don't let them type more than one though: **12.3.56.5.** It is hard to do math on these numbers. (disable the decimal button if there's already one in the display)
h) Advanced: Add a "backspace" button, so the user can undo if they click the wrong number.
i) Advanced: Add keyboard support!