Week 1 - Fundamentals

Overview
For this week, you will be familiarizing yourself with the basic constructs of programming: loops, conditionals, logic operators, and a few techniques.

Prerequisites
It is imperative that you complete the 13 mandatory algorithm challenges from earlier in the bootcamp.

Study Guide
Here is a list of concepts to study. Some or all of these will be used to solve this week's challenges.

variables, functions *for loops, while loops*
conditional (if / else) statements
console.log *return values*
math.random, math.floor, math.ceil
&& || ! (and, or, not) *% (modulus)*

T-Diagrams
Being able to write a t-diagram to keep track of your variables while you write out an algorithm by hand is extremely beneficial. You should use a t-diagram for every algorithm challenge this week.

TDD , Tests, and Examples
To put it simply. test-driven development (TDD) is a design technique where you must **first write a test that fails** before you write any new code, with the **goal of writing clean code that passes the test**. We will supply tests, if/else checks, and sample input/outputs to help guide your solutions.

Extra Goodies
At the bottom of the page are some examples of simple Javascript constructs we'll use this week. Remember these basic building blocks!

For Loops

```
for (INITIALIZATION; TEST; INCREMENT/DECREMENT)
{
    // body of the loop -- run repeatedly while TEST is true
    // INIT.   TEST?-BODY-INCREMENT.   TEST?-BODY-INCREMENT.   TEST?-[exit]
}
```

Conditional (if / else) statements

```
if (CONDITION_1 && CONDITION_2)
{
    // body of the 'if' statement -- only runs if CONDITION_1 is true AND CONDITION_2 is true
}
else
{
    // body of the 'else' statement -- only runs if CONDITION_1 is false or CONDITION_2 is false
}
```

Functions

```
// Declaring standalone functions
function MY_FUNCTION(PARAMETER_1, PARAMETER_2)
{
    // body of the function -- only runs if function is invoked
}

// Calling standalone functions. ARGUMENTS passed by a caller enter the function as PARAMETERS
MY_FUNCTION(ARGUMENT_1, ARGUMENT_2);
```

Tomorrow: *sum fun*? Sigma and factorial

Week 1 - Fundamentals - '13' Review #1

Print 1-255

Print all the integers from 1 to 255.

```
function print1to255()
{
  var num = 1;
  while (num <= 255) {
    console.log(num);
    num = num + 1;
  }
}
```

Print Odds 1-255

Print all odd integers from 1 to 255.

```
function printOdds1to255()
{
  var num = 1;
  while (num <= 255) {
    console.log(num);
    num = num + 2;
  }
}
```

Print Sum 0-255

Print integers from 0 to 255, and the sum so far.

```
function printSum1to255()
{
  var sum = 0;
  for (var num = 0; num <= 255; num++) {
    sum += num;
    console.log("New number:", num,
                "Sum:", sum);
  }
}
```

Iterate Array

Print all values in a given array.

```
function printArrayValues(arr)
{
  for (var index = 0;index < arr.length;index++)
  {
    console.log("array[", index,
                "] is equal to", arr[index]);
  }
}
```

Find Max

Print the largest element in a given array.

```
function printArrayMax(arr)
{
  if (arr.length == 0) {
    console.log("Empty array, no max value.");
    return;
  }
  var max = arr[0];
  for (var idx = 1; idx < arr.length; idx++) {
    if (arr[idx] > max) {
      max = arr[idx];
    }
  }
  console.log("Max value is:", max);
}
```

Get Average

Analyze an array's values and print the average.

```
function printArrayAverage(arr)
{
  if (arr.length == 0) {
    console.log("Null arr, no average val");
    return;
  }
  var sum = arr[0];
  for (var idx = 1; idx < arr.length; idx++) {
    sum += arr[idx];
  }
  console.log("Average value is:",
              sum / arr.length);
}
```

Tomorrow: *sum fun*? Sigma and factorial