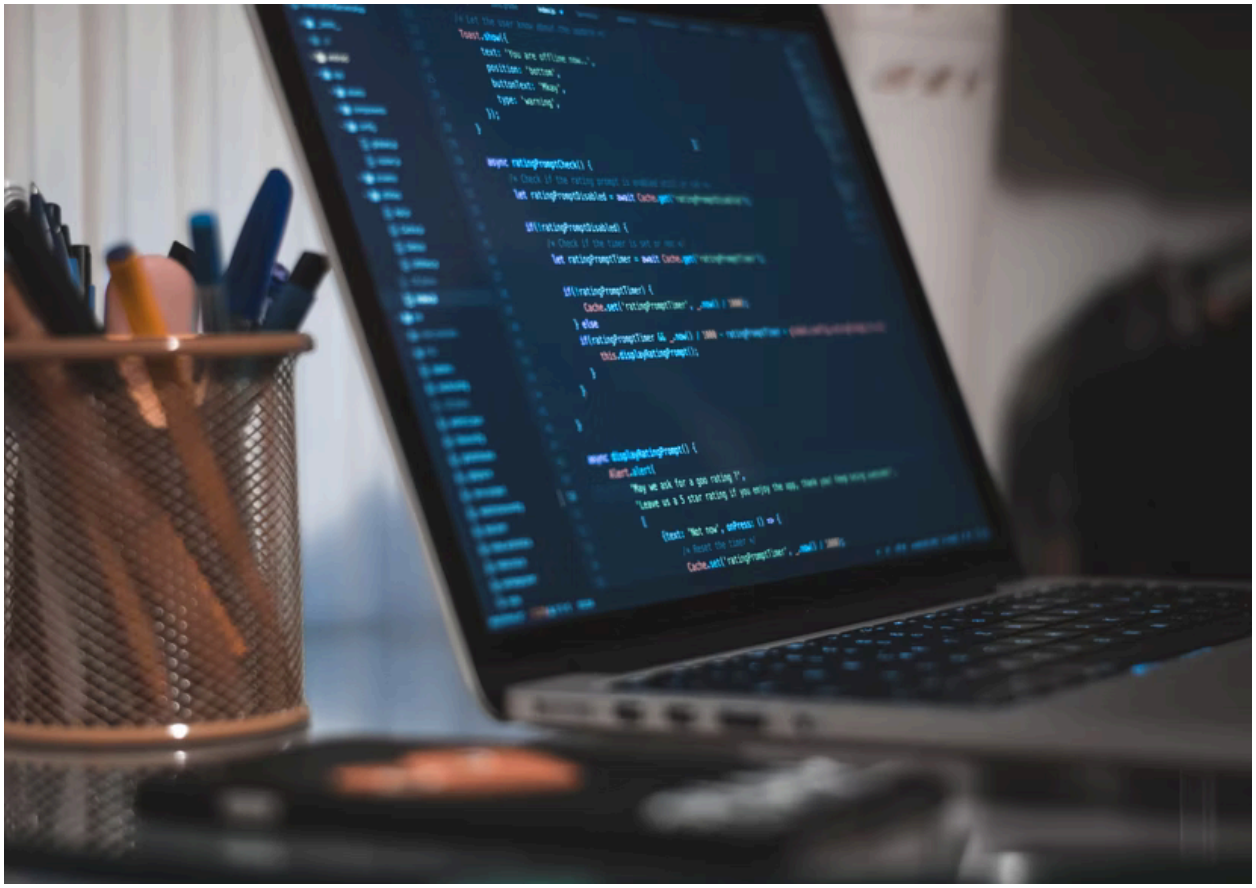

JavaScript Practice Challenges

Problem-Solving.

Project description.

Use your knowledge of JavaScript to work through the code challenges given in this document. **Do these code challenges individually**. Aim to have a solution before you google this will help you learn quite a lot and also help you reinforce your problem-solving skills in JavaScript.

You can use Vanilla JavaScript to complete these code challenges. Vanilla JS is recommended because that is what the lessons have covered so far and you should get some practice with it.

Caution:

- **DO NOT Copy solutions online or from anywhere**.
  Try to do this yourself

  Prerequisites.
- It would be best if you had covered the JS basics (Variables, Control flow, Arrays, and Objects).
- Personal research and prior practice with JavaScript will help you a lot.
- You should **create a GitHub repo** for these challenges. Then create separate files for each solution of a code challenge.

Submission deadline:

Code Challenges (Set 1)

—----------

1. Write a function named **fizzBuzz** that takes in two(2) parameters which are expected to be strings. The function should return the string **Fizz** if the combined length of the parameters is divisible by 3, the string **Buzz** if it is divisible by 5, and the string **FizzBuzz** if it is divisible by both 5 and 3.

2. Write a JavaScript program that **prompts** a user to enter their year of birth and in turn prints a string in the console stating whether the user is a **minor, a youth, or an elder**. Anyone **below 18 years is a minor**, anyone **between 18 and 36 years is a youth** and the rest are elders.

3. Write a function named **twoSum** which takes two parameters: nums and target. Given an array of integer **nums** and an integer **target**, return indices of the two numbers such that they add up to the **target**. You may assume that each input would have exactly one solution, and you may not use the same element twice.

   You can return the answer in any order.

   Example 1:

   ```
   Input: nums = [2,7,11,15], target = 9
   Output: [0,1]
   Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
   ```

   Example 2:

   ```
   Input: nums = [3,2,4], target = 6
   Output: [1,2]
   ```

4. Write a function named **firstPalindrome** which takes a parameter: **words**. Given an array of string **words**, return the first palindromic string in the array. If there is no such string, return an empty string "".

   A string is palindromic if it reads the same forward and backward.

   Example 1:

   ```
   Input: words = ["abc","car","ada","racecar","cool"]
   Output: "ada"
   Explanation: The first string that is palindromic is "ada".
   Note that "racecar" is also palindromic, but it is not the first.
   ```

5. Given an integer **num**, write a function that repeatedly adds all its digits until the result has only one digit, and return it.

Example 1:

```
Input: num = 38
Output: 2
Explanation: The process is
38 --> 3 + 8 --> 11
11 --> 1 + 1 --> 2
Since 2 has only one digit, return it.
```