CMPS 350 Web Development Fundamentals
Lab 5 – JavaScript Fundamentals

Objective

The objective of this Lab is to practice the following JavaScript topics.

- ✓ **Basics of JavaScript**
- ✓ **Control Structures**
- ✓ **Functional Programming and Higher-Order Functions**
    - **Arrow function**: allows shorter syntax for writing **functions**.
    - **Array functions** (.map, .reduce, .filter, flat,. Splice, .sort…)
    - **Spread operator**

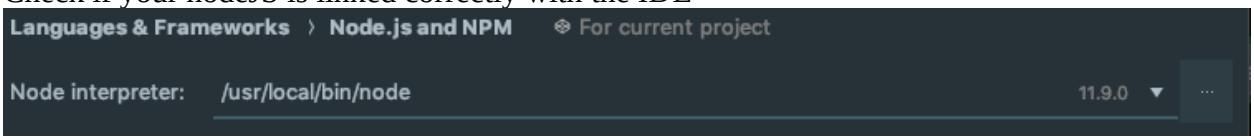
Overview

This Lab has three parts:

- ❖ **Exercise 1**: Has some warm-up exercise to help you practice basics Javascript syntax
- ❖ **Exercise 2:** Helps you practice Arrays and control structures
- ❖ **Exercise 3: Help you practice** arrow functions and array functions

Preparation

1. If you have not installed nodeJS on your laptop then visit this site and install the appropriate nodejs for your platform https://nodejs.org/en/download/
2. Go to
    - ➢ File | Settings | Languages and Frameworks | Node.js and NPM for Windows and Linux
    - ➢ WebStorm | Preferences | Languages and Frameworks | Node.js and NPM for macOS
3. Check if your nodeJS is linked correctly with the IDE



4. Press OK and restart your IDE
5. Now in the terminal type "node -v" you should see the version of the nodejs you have installed. The version should be 14+

```
(base) Abdulahis-MBP:~ abdulahihassen$ node -v
v14.13.0
```

Exercise 1 – Basic Syntax Warm up JS exercises

1. Create a JavaScript file named **excercise1.js** inside the Lab 5-JS folder
2. Create a file called app.js inside the project folder
3. Using a ***While*** loop, write a JavaScript program that displays odd numbers from 1 to 100.
4. Rewrite the first program using a ***For*** loop.
5. Consider the following array declaration: *let cars = ["Saab", "Volvo", "BMW"];*
    - Add "**Toyota**" to the end of the array
    - Add "**Mercedes**" to the beginning of the array.
    - Create a ***displayArray*** function that takes an array as an argument and prints the array elements individually using a ***For-of*** loop. Call the "***displayArray*** " function to display the cars array.
    - Sort the array alphabetically and print it again.

Exercise 2 – Arrays

1. Create a JavaScript file named **excercise2.js** inside the Lab 5-JS folder
2. Check the following tutorial on **ECMAScript6 -** https://www.w3schools.com/js/js_es6.asp
3. Create an array that contains the following numbers [2,4,18,28,9,5,6,7,8,9] and name it **dataPoints**;
4. Implement the following functions and use one of the Console functions (.log , .error , .trace , .table , .time , .endTime) to display your results.
    a. Display all the elements in the dataPoints array by using For..of and forEach
    b. Remove the first two elements from the **dataPoints** array
    c. Remove the last two elements from the **dataPoints** array
    d. Add 10 and 12 to the **dataPoints** array
    e. Delete the largest number from the **dataPoints** array
    f. Sort the elements in the **dataPoints** array in both ascending and descending orders
    g. Find the average of the **dataPoint** array
    h. Create a second array named **newPoints** which contains the following values 55,66,77,88,99.
    i. Combine the two arrays into a single array. You should add the **newPoints** array to the **dataPoints** array and save them to an array called "**combinedDataPoints**"

j. Find the sum of all the elements inside **combinedDataPoints** array

k. Find the maximum and the minimum numbers in the **combinedDataPoints** array

Exercise 3 – ES6 - Higher-Order Functions

1.
```
let matrix = [ [2, 3], [34, 89], [55, 101, 34], [34, 89, 34, 99]];
```

Use the above array and Implement and test the following functions:

- **flatten:** gets a matrix (i.e., array of arrays) and returns a single dimensional flat array.
- **max**: gets an array and returns its maximum value. Use reduce function.
- **sort:** gets an array and returns a sorted array in descending order (from big to small).
- **square:** gets an array and returns an array with squared values.
- **average:** gets an array and returns its average.
- **removeDuplicate:** gets an array and returns an array without duplicate elements.
- **Filter :** Find the sum of all the number in the array that are greater than 40. You should write everything as one single statement.

Use the following matrix to test your work.

Expected output:

```
Original array:
[ [ 2, 3 ], [ 34, 89 ], [ 55, 101, 34 ], [ 34, 89, 34, 99 ] ]

Flattened:
[ 2, 3, 34, 89, 55, 101, 34, 34, 89, 34, 99 ]

Max value:
101

Sorted in descending order:
[ 101, 99, 89, 89, 55, 34, 34, 34, 34, 3, 2 ]

Without duplicate elements:
[ 101, 99, 89, 55, 34, 3, 2 ]

Sum of unique elements:
574

Square of unique elements:
10201 9801 7921 7921 3025 1156 1156 1156 1156 9 4
```