COMP1231
Web Programming

Assignment #3

Due Date: Friday, July 22nd, 11:59 pm

Weight: 8% of Final Grade

Table of Contents

COMP1231 Assignment 3 3
Function 1: Swap Characters 4
Function 2: Move Capital Letters 5
Function 3: Repeating Characters 6
Function 4: Capitalize First Letter of Each Word 7
Function 5: Remove Duplicates 8
Submission Procedure and Rules 9
  Submission Procedure 9
Marking Rubric 10

COMP1231 Assignment 3

Assignment Type: Individual Assignment

1. **Description**
   In this assignment, you are to develop a JavaScript file (problems.js) that contains the implementation of **5 functions**. Each function is represented as one step, and marks will be awarded on the completion of each step (each function).

   Each function is independent and solves a unique problem, as such, treat and implement each function in isolation of the others, that is, you should only focus on one problem at a time.

   PLEASE NOTE: Unless indicated within the question(s), input validation is NOT required for a functions input parameter – You may assume that all input parameters provided to your functions, are valid.

2. **Objective**
   - Write decision-making statements and control structures to solve problems
   - Apply programming logic to solve basic to intermediate problems
   - Testing and debugging

3. **Learning Outcomes:** After conducting this assignment students will / will be able to:

   1. Acquainted with implementing JavaScript functions
   2. Follow instructions, and to implement various requirements
   3. Acquainted with authoring JavaScript unit tests to validate their functions

4. **Instructions:**

   1. You are to create ONE JavaScript file which must be named **<*STUDENT_ID*>-problems.js**, where ***STUDENT_ID*** represents your student number.

   2. The implementation for your **5** JavaScript functions must be completed within your **<*STUDENT_ID*>-problems.js** file.

      You are **NOT** permitted to use any external files or libraries. Using any external libraries will result in a final grade of ZERO for your assignment submission.

   3. Your JavaScript functions file **must** be configured for strict mode (“**use strict**”)

The pages that follow provide the description of the **5** JavaScript functions that you must implement.

Function 1: Swap Characters

Create a JavaScript Arrow function that meets the following requirements:

- Authored using **arrow expression** syntax (constant name is **_swapCharacters**)
- That take three arguments, a string, character 1 and character 2
- The function replaces all instances of c1 with c2, and vice versa
- The function returns the updated string
- Console log output is NOT permitted.
- The function should pass each of the illustrated examples below at a minimum.

```
_ swapCharacters( "aabbccc", "a", "b")                    ➔ "bbaaccc"
_ swapCharacters("random w#rds writt&n h&r&","#","&") ➔ "random w&rds writt#n h#r#"
_ swapCharacters("128 895 556 788 999", "8", "9")       ➔ "129 985 556 799 888"
```

Function 2: Move Capital Letters

Create a JavaScript Arrow function that meets the following requirements:

- Authored using **arrow expression** syntax (constant name **_moveCapitalLetters)**
- That takes in a string parameter, of mixed casing (mix of upper and lowercase letters)
- The function moves all capital letters to the front of a word.
- The uppercase letters moved to the front, maintain their original relative order
- The lowercase letters moved to the back front, maintain their original relative order
- Console log output is NOT permitted.
- The function should pass each of the illustrated examples below at a minimum.

```
_moveCapitalLetters("hApPy")       ➔ "APhpy"
_moveCapitalLetters("moveMENT")    ➔ "MENTmove"
_moveCapitalLetters("shOrtCAKE")   ➔ "OCAKEshrt"
```

Function 3: Repeating Characters

Create a JavaScript Arrow function that meets the following requirements:

- Authored using **arrow expression** syntax (constant name **_repeatingCharacters)**
- That takes in a string
- The function returns the first character that repeats
- If there is no character that repeats, return -1
- The function should be case sensitive (ex: "I" not equal to "i")
- Console log output is NOT permitted.
- The function should pass each of the illustrated examples below at a minimum.

```
_repeatingCharacters("legolas")   ➔ "l"
_repeatingCharacters("Gandalf")  ➔ "a"
_repeatingCharacters("Balrog")   ➔ "-1"
_repeatingCharacters("Isildur")  ➔ "-1"
```

Function 4: Capitalize First Letter of Each Word

Create a JavaScript **function expression** that meets the following requirements:

- Authored using **function expression** syntax (with constant name **_capitalizeFirstLetter)**
- That takes in a string as an argument
- The function converts first character of each word to uppercase
- The function returns the newly formatted string
- Console log output is NOT permitted.
- The function should pass each of the illustrated examples below at a minimum.

```
_capitalizeFirstLetter("This is a title")        → "This Is A Title"
_capitalizeFirstLetter("capitalize every word")  → "Capitalize Every Word"
_capitalizeFirstLetter("I Like Pizza")           → "I Like Pizza"
_capitalizeFirstLetter("PIZZA PIZZA PIZZA")      → "PIZZA PIZZA PIZZA"
```

Function 5: Remove Duplicates

Create a JavaScript function expression that meets the following requirements:

- Authored using **function expression** syntax (with constant name **_removeDuplicates)**
- That takes an array of items (numbers or strings) as argument
- The function removes all duplicate items in the array
- The function returns a new array in the same sequential order as the original source array (minus the duplicates)
- Console log output is NOT permitted.
- The function returns an array containing the array groupings back to the caller.

```
_removeDuplicates([1, 0, 1, 0])              ➔ [1, 0]
_removeDuplicates(["The", "big", "cat"])     ➔ ["The", "big", "cat"]
_removeDuplicates(["John", "Taylor", "John"]) ➔ ["John", "Taylor"]
```

Submission Procedure and Rules

Please ensure to remove all instances of the following from your final submission solution:

- document.write()
- innerHTML()
- alert()
- any commented code
- Do not over use the console log, spamming the console log unnecessarily, say for example. with debug related output may cost you marks.

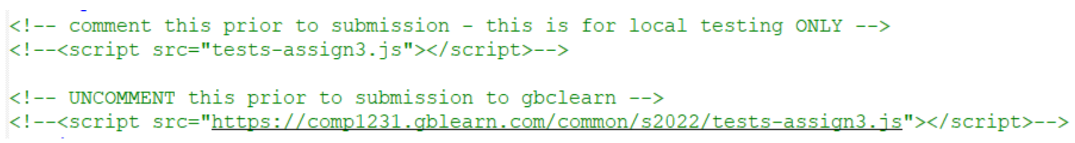
Submission Procedure

1. In order to complete this assignment, you will need to create the following **TWO** files.

   I. ***<STUDENT_ID>*-problems.js**
   The problem.js file (template provided), as mentioned in the assignment instructions and demonstrated in class, the <STUDENT_ID>-problems.js is where your functions will be implemented. A template of this file has been provided to you. The <student_id>-problems.html is a compulsory component of your submission.

   II. **index.html**
   The index.html file (template provided) visually displays the result of your unit tests and was also demonstrated in class. You must refactor the index.html provided to properly reference your <student_id>-problems.js. The index.html is a compulsory component of your submission.

   PLEASE NOTE!!
   For your index.html, please make sure to **UNCOMMENT** the path to the remote test.js file to allow for remote testing, and **COMMENT** the path to the local test.js (when you are complete local testing). Comment within the HTML have been provided to make the process trivial for you.



```
<!-- comment this prior to submission - this is for local testing ONLY -->
<!--<script src="tests-assign3.js"></script>-->

<!-- UNCOMMENT this prior to submission to gbclearn -->
<!--<script src="https://comp1231.gblearn.com/common/s2022/tests-assign3.js"></script>-->
```

2. The **tests.js** file (template also provided) represents the unit test file and was also demonstrated in class. You are encouraged to use this file as a means to author suitable unit tests that validate the health of your functions. The tests.js is **NOT** part of your submission, so please **DO NOT** submit this.

3. You must upload both your files (<student_id>-problem.js and index.html) to the following directory in your **GBlearn** account on or before the due date: /comp1231/assignments/assignment3/

4. You must also submit your JavaScript file on **my.gblearn.com** under **comp1231/assignment 3**.

   Avoiding this step will result in a mark of ZERO for your assignment submission. All files/folders must be spelled exactly as specified above (all lowercase without any spaces)

5. Late submissions are graded at a **-10%** penalty per day (five day maximum)

We are going to use Moss (https://theory.stanford.edu/~aiken/moss/) to detect similarities in code among all students. There will be zero tolerance for plagiarism. Please be aware.

Marking Rubric

40% - Your overall code and functionality
40% - Logic
20% - Best practices

- Variables/function name can be short, yet descriptive/meaningful, camelCasing applied
- Use let/const over keyword var when possible
- Avoid globals
- Stick to a strict coding style
- Comment as much as needed but not more
- Clean and easy to read code (indentation, space before and after any operator)

GOOD LUCK!