CISC 260 Assignment 2

Due Date: February 18, 2016 before midnight

1 Description

In this assignment, we will be practicing to write some JavaScript code. You will be writing few functions whose details are as follows:

1. Write a program that uses **console.log** to print all the numbers from 1 to 100, with two exceptions. For numbers divisible by 3, print **"Fizz"** instead of the number, and for numbers divisible by 5 (and not 3), print **"Buzz"** instead.

   When you have that working, modify your program to print **"FizzBuzz"**, for numbers that are divisible by both 3 and 5 (and still print **"Fizz"** or **"Buzz"** for numbers divisible by only one of those).

2. You can get the Nth character, or letter, from a string by writing **"string".charAt(N)**, similar to how you get its length with **"s".length**. The returned value will be a string containing only one character (for example, "b"). The first character has position zero, which causes the last one to be found at position **string.length - 1**. In other words, a two-character string has length 2, and its characters have positions 0 and 1.

   Write a function **countBs** that takes a string as its only argument and returns a number that indicates how many uppercase "B" characters are in the string.

   Next, write a function called **countChar** that behaves like **countBs**, except it takes a second argument that indicates the character that is to be counted (rather than counting only uppercase "B" characters). Rewrite **countBs** to make use of this new function.

3. Arrays have a method **reverse**, which changes the array by inverting the order in which elements appear. For this exercise, write two functions, **reverseArray** and **reverseArrayInPlace**. The first, **reverseArray**, takes an array as argument and produces a new array that has the same elements in the inverse order. The second, **reverseArrayInPlace**, does what the reverse method does: it modifies the array given as argument in order to reverse its elements. Neither may use the standard revers method.

4. The == operator compares objects by identity. This means if two objects point to the same memory location it would return true otherwise false. However, sometimes, you would prefer to compare the values of their actual properties.

   Write a function, **deepEqual**, that takes two values and returns true only if they are the same value or are objects with the same properties whose values are also equal when compared with a recursive call to **deepEqual**.

   To find out whether to compare two things by identity (use the === operator for that) or by looking at their properties, you can use the **typeof** operator. If it produces **"object"** for both values, you should do a deep comparison. Do you have to take any exception into account?

1.1 Submission

Save all your .js files only, in a folder named <**your firstname** >_ <**your lastname** >. Compress this folder as **.zip** file and submit it through Moodle. Please do not dump any other files in this folder. You are required to submit only .js file(s) and nothing else.

Note: Submitting wrong files or in the wrong format will not be accepted nor will any re-submission be allowed for any such mistake.