JavaScript Basics Assignment

***1. Find the smallest number in an array***

*Create a function that will display the smallest value in the array.*

**Example:**

> console.log(findSmallest([30, 45, 60, 7]));

> 1

**Reference:**

* [Math.min](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/min)

*2. Sort strings by Alphabetical Order*

*Function that will return your string in Alphabetical order*

**Example:**

> console.log(AlphabeticalOrder('hello'));

> "ehllo"

**Reference:**

* [Array.sort](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort)
* [Array.split](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split)
* [Array.join](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/join)

***3. Factorialize a number***

*In mathematics, the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n.* *In simple terms, the Factorial of 7 is solved like this:*

**7 \_ 6 \_ 5 \_ 4 \_ 3 \_ 2 \_ 1 = 5,040**

**Example:**

> console.log(factorializer(7));

> 5040

**Reference:**

* [What is Factorial?](https://en.wikipedia.org/wiki/Factorial)
* [Recursion](https://www.youtube.com/watch?v=k7-N8R0-KY4)
* [If and Else statements](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/if...else)

***4. Identify if a number is Odd or Even?***

*A function that lets you know if a number is Even or Odd*

**Example:**

> console.log(oddOrEven(7));

> "Odd"

**Reference:**

* [Modulo](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Arithmetic_Operators#Remainder_())

**5. Eliminate all odd numbers in an array.**

Remove all Odd number(s) in an array and return a new array that contains Even numbers only

**Example:**

> console.log(evenOnly([1, 2, 3, 4, 5, 6]));

> [ 2, 4, 6 ]

**Reference:**

* [Array.filter](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter)
* [Modulo](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Arithmetic_Operators#Remainder_())

**6. Return numbers only**

Create a function that will accept an array, check the data type of each element. The function will delete string elements and will return a the new array

**Example:**

> console.log(numbersOnly(['text', 3, 7, 'github', 13, 'dev']));

> [ 3, 7, 13 ]

**Reference:**

* [Array.filter](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter)
* [typeof](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/typeof)

**7. Add up the numbers**

Return the sum of a number going back to it's root. In other words, the function will work like this:

**addUp(5);**

// 5 + 4 + 3 + 2 + 1 + 0 = **15**

**Example:**

> console.log(addUp(8));

> 36

**Reference:**

* [Recursion](https://www.youtube.com/watch?v=k7-N8R0-KY4)
* [1 + 2 + 3 + 4 + ⋯](https://en.wikipedia.org/wiki/1_%2B_2_%2B_3_%2B_4_%2B_%E2%8B%AF)

**8. Return the Min, Max, Length and Average of an Array**

Create a function that will accept an array and do the following:

* Get the lowest element
* Get the highest element
* Get the length of array
* Get the Average of all element;
* Store these criteria in a new array

**Example:**

> console.log(minMaxLengthAverage([7, 13, 3, 77, 100]));

> [ 3, 100, 5, 40 ]

**Reference:**

* [Math.min](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/min)
* [Math.max](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/max)
* [Array.reduce](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/reduce)
* [Array.length](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/length)

***9. Sort Numbers in Ascending Order***

*Array.sort()* sorts the ***strings*** alphabetically. What if we want to sort ***numbers*** from lowest to highest? Will it produce a correct output?

**Example:** This is what happen if we apply *Array.sort()* to numbers:

> arr = [45, 34, 23, 12, 7]

> console.log(arr.sort());

> [ 12, 23, 34, 45, 7 ]

Output is not correct right?, whereas we are expecting this to be the return value:

> console.log(sortNumsAscending([7, 13, 3, 77, 100]));

> [ 3, 5, 40, 100 ]

**Reference:**

* [Sorting in JavaScript](http://www.javascriptkit.com/javatutors/arraysort.shtml)
* [Array.sort()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort)

**10. Convert Numbers in Roman Numerals**

Convert the given number to a Roman Numeral

**Example:**

> romanNumbers(1989);

> MCMLXXXIX

**Reference:**

* [Roman Numerals](https://www.mathsisfun.com/roman-numerals.html)
* [Array.join()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/join)
* [Array.indexOf()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/indexOf)
* [Array.splice()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice)

**11. Absolutely Sum**

Return the absolute sum of all the array elements

**Example:**

> getAbsSum([-1, -3, -5, -4, -10, 0]);

> 23

**Reference:**

* [Array.prototype.reduce](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/Reduce)
* [Math.abs](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/abs)

***12. Looping a Triangle***

*Form a triangle using hash tags*

**Example:**

> #

> ##

> ###

> ####

> #####

> ######

> #######

**Reference:**

* [Loop](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration)

**13. Count the number of Words**

Return how many words was given

**Example:**

> countWords('hello from kbpsystem!');

> 3

**Reference:**

* [String.prototype.split](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split)
* [Array.length](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/length)

**14. Multiply by Length**

Multiply all elements in an array by it's length

**Example:**

> MultiplyByLength([4,1,1]);

> [12, 3, 3]

**Reference:**

* [for](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/for)
* [Array.length](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/length)
* [Array.prototype.push](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/push)
* [Array.prototype.map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map)

**15. Repeating Letters**

Create a function that will repeat each string character two times

**Example:**

> console.log(doubleChar('exercise'));

> eexxeerrcciissee

**Reference:**

* [Array.prototype.split](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split) The split() method splits a String object into an array of strings by separating the string into substrings, using a specified separator string to determine where to make each split.
* [Array.prototype.map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map) The map() method creates a new array with the results of calling a provided function on every element in the calling array.
* [Array.prototype.join](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/join) This method joins all elements of an array (or an array-like object) into a string and returns this string.

# Array Functions Assignment

16. Create a function named reversePlusOne. This function should:

* + Take one argument, an array of at least two numbers.
  + This function should return:
    - the array *reversed* with a 1 added at the front

For example:

reversePlusOne([1,2]); // returns [1,2,1]

reversePlusOne([5,4,3,2]); // returns [1,2,3,4,5]

* Create a function named plusesEverywhere. This function should:
  + Take one argument, an array of at least two numbers.
  + This function should return:
    - a String made of all the values in the array separated by +

For example:

plusesEverywhere([1,2,3]); // returns "1+2+3"

plusesEverywhere([18,24]); // returns "18+24"

* Create a function named arrayQuantityPlusOne. This function should:
  + Take one argument, an array of numbers.
  + This function should return:
    - one greater than the number of items in the array

For example:

arrayQuantityPlusOne([0,0,1,0,2,1]); // returns 7

arrayQuantityPlusOne([42]); // returns 2

# Object Oriented Basics

17. Complete the createCourse function. This function should:

* + take three arguments that will define course properties
    - courseTitle (string)
    - courseDuration (string)
    - courseStudents (array)
  + return an object that has each property assigned its proper value

For example:

createCourse('Bloc Front-End Engineering', '4 weeks', ['Joe', 'Tim', 'Rob'])

// should return {title: 'Bloc Front-End Engineering', duration: '4 weeks', students: ['Joe', 'Tim', 'Rob']}

* Complete the addProperty function. This function should:
  + Take three arguments:
    - object: an object to add a properties to
    - newProp: a property that we want to add to the object
    - newValue: a value that we want the new property to have
  + If object doesn't already have a property named newProp, then add newProp with value of newValue to object
  + If object already has newProp, return the object argument.

For example:

addProperty({}, 'firstName', 'Jim') // should return { firstName: 'Jim' }

addProperty({firstName: 'Rob'}, 'firstName', 'Jim') // should return {firstName: 'Rob'}

* Complete the formLetter function. This function should:
  + take one argument, a letter, which has three properties recipient, sender, and msg
  + combine the three properties into a single string with an additional greeting and closing
  + insert additional new lines between the greeting, message, and signature.

For example:

formLetter({ recipient: "James", sender: "Richard", msg: "Things are well." })

// should return "Hello James,\n\nThings are well.\n\nSincerely,\nRichard"

* Complete the canIGet function. This function should:
  + Take two arguments:
    - item: represents what the user wants to buy
    - money: represents how many dollars a user has
  + return true if a user can afford a given item according to the price chart below, and false otherwise:
    - 'MacBook Air' - $999
    - MacBook Pro' - $1299
    - 'Mac Pro' - $2499
    - 'Apple Sticker' - $1
  + Return false if the item is not in the above list of Apple products

Do this with 0 'if' conditions! (Hint: Place the above price table in an object).

For example:

canIGet('MacBook Air', 100) // returns false

canIGet('MacBook Air', 1000) // returns true

# Strings Assignment

We'll be working on five functions for this exercise.

18. Complete the formLetter function. This function should:

* + Take three strings as arguments: the first name of the recipient, the sender's signature name, and the message of the letter
  + combine the three into a single string with additional greetings and closings
  + insert additional new lines between the greeting, message, and signature

For example:

Ex.formLetter("James", "Richard", "Things are well.");

...should return:

"Hello James,\n\nThings are well.\n\nSincerely,\nRichard"

* Complete the sliceItAndCombineIt function. This function should:
  + take a string and four indices (numbers)
  + return a new string which is the concatenation of two substrings marked by the first and second index of each pair of indices. For example:

sliceItAndCombineIt("This is a Test", 0, 4, 5, 7) // returns "Thisis"

sliceItAndCombineIt("This is a Test", 0, 4, 1, 2) // returns "Thish".

* Complete the findFirstMatch function. This function should:
  + Take two strings as arguments. The first string is the one to search, the second is the one to search for.
  + Return the position (i.e. index) of the first match of string being searching for

For example:

findFirst("Roses are red", "re") // returns 7 (the position of the "re" in "are")

* Complete the findLastMatch function. This function should:
  + Take two strings as arguments. The first string is the one to search, the second is the one to search for
  + Return the position (a.k.a. the index) of the last match of string we're searching for For example:

findFirst("Roses are red", "re") returns 10 (the position of the "re" in "red")

* Complete the substringBetweenMatches function. This function should:
  + Take two strings as arguments. The first string is the one to search, the second is the one to search for
  + Return the substring between the first match and the last match
  + Not include the first match or the last match in the returned substring For example:

findFirst("Roses are red, apples are really red.", "red") // returns ", apples are really "

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@