# Greater than 50

Write a function that takes a number and:

* returns Correct! if the number is over 50.
* returns Incorrect! if the number is less than or equal to 50.
* returns an empty string if the number is zero or less.

//RETURN ORDER, WORKING

function greaterThan50(number) {

* if (number <=0){
* return ("");
* }
* if (number>50){
* return ('Correct!');
* }
* if (number<=50) {
* return ('Incorrect!');
* }
* }

**Exclaim Word WORKING**

This function will be called with a string and should return the string in upper case with an exclamation mark (!) added to the end of the string.

Examples:

exclaimWord('bang');

// returns BANG!

exclaimWord('cool');

// returns 'COOL!'

// please complete this function...

function exclaimWord(word) {

  //Upper case the string

  //Add an ! mark at the end

let output =word.toUpperCase();

return (output); or below

return(word.toUpperCase + '!')

}

# Create Sentence WORKING

This function should take an object with three properties: name, age and job.

It should return a string in the format shown below:

"Hello my name is <name>, I am <age> years old and I am a <job>".

Examples:

createSentence({ name: 'Mitch', age: 29, job: 'tutor' });

// should return "Hello my name is Mitch, I am 29 years old and I am a tutor"

createSentence({ name: 'Frankie', age: 31, job: 'software engineer' });

// should return "Hello my name is Frankie, I am 31 years old and I am a software engineer"

// please complete this function...

function createSentence(person) {

  //3 property keys

  //Create string with object key value properties and concantenante  them

 let output = "Hello my name is " + person.name + ", " +  "I am " + person.age + " years old"+  " and I am a " + person.job

//Learn how to do this as a string litteral

   return (output)

}

# Get First and Last WORKING

This function should take an array and return a new array containing the first and last item from the input array.

The function should return the input array when passed an input array with fewer than 2 items.

getFirstAndLast(['a', 'b', 'c', 'd', 'e', 'f']);

// should return ['a', 'f']

getFirstAndLast([42, 100, 25, -3]);

// should return [42, -3]

getFirstAndLast([10, 50]);

// should return [10, 50]

getFirstAndLast([5]);

// should return [5]

// please complete this function...

# getFirstAndLast(['a', 'b', 'c', 'd', 'e', 'f']);

# // should return ['a', 'f']

# fetFirstAndLast([42, 100, 25, -3]);

# // should return [42, -3]

# getFirstAndLast([10, 50]);

# // should return [10, 50]

# getFirstAndLast([5]);

# //should return [5]

# function getFirstAndLast(array) {

# //Create a new array containing only the first an last items of the array

# //let output=[];

# let output = array.splice(0,1)

# let output2 =array.splice(-1,1)

# let final = output + ',' + output2;

# console.log(final);

# return (final);

# }

# Find Total Age WORKING

This function will be called with an array of objects. Each object represents a person and will contain an age property. The function should return the total age of all the people in the array.

A typical people array will look as follows:

[

{

name: 'Sam',

age: 30,

},

{

name: 'Anat',

age: 22,

},

{

name: 'Jonny',

age: 32,

},

];

Examples:

findTotalAge([]); // should return 0

findTotalAge([{ name: 'Sam', age: 31 }]); // returns 31

findTotalAge([

{ name: 'Sam', age: 20 },

{ name: 'Anat', age: 30 },

{ name: 'Alex', age: 40 },

]);

// returns 90

// please complete this function ...

function findTotalAge(people) {

  //Array of objects, Age property

  //Sum the age key values of the array nof objects

  let total =0;

for (let i = 0; i <= people.length - 1; i++) {

      if (people[i].age >0)

      total = total +people[i].age

      }

      console.log(total)

      return (total);

 }

# Count the Characters DONE

Complete the function countChars which takes a string and a character and returns the number of times this character is found within the string.

Think about how you will access each character of the input string in order to test whether it is the character you are looking for.

The string will only contain lowercase letters and the character you are being asked to count will always be lowercase.

countChars('hello', 'l') --> 2

countChars('hello world', 'l') --> 3

countChars('hello world', 'z') --> 0

countChars('hello world', 'e') --> 1

function countChars (str, char) {

//Create a variable that equals the value of the regEx function, //which includes the char parameter passed from the main function,

//and the g switch for global, thus search for more than one occurance

let regEx = new RegExp(char, 'g');

//Return the total number of occurrences of the match function to a //number.length

let charCount = (str.match(regEx) || []).length;

return (charCount);

}

countChars("hello my name is Jon", "J")

# All Oscar Winners WORKING

This function will be called with an array of film objects. It should check that every film in the array has won an oscar.

If there is at least one film where the wonOscar property is false then the function should return false overall. Otherwise, if every film has won an oscar then the function should return true.

A typical array of films will look as follows:

[

{

title: 'Forrest Gump',

wonOscar: true,

},

{

title: 'Moonlight',

wonOscar: true,

},

{

title: 'No Country for Old Men',

wonOscar: true,

},

];

Examples:

allOscarWinners([{ title: 'Forrest Gump', wonOscar: true }]);

// should return true;

allOscarWinners([{ title: 'Ghost World', wonOscar: false }]);

// should return false

allOscarWinners([

{

title: 'Forrest Gump',

wonOscar: true,

},

{

title: 'Moonlight',

wonOscar: true,

},

{

title: 'No Country for Old Men',

wonOscar: true,

},

]);

// should return true

allOscarWinners([

{

title: "Forrest Gump", wonOscar: true

},

{

title: "Moonlight", wonOscar: true

},

{

title: "The Breadwinner", wonOscar: false

}

{

title: "No Country for Old Men", wonOscar: true

}

]

);

// should return false

// please complete this function ...

function allOscarWinners(films) {

//Array of fim object

//Check for Oscar win property true

//If wonOscar = false return false, else true

//let boolean = films.reduce((key, property) => (property.wonOscar === "true"  //? key + false : key),   true );

let counter =0;

//need to iterate through the object key properties

for (let i = 0; i < films.length; i++) {

      if (films[i].wonOscar === false) {

       return false

      }

    }

   return true

    }

# Gather Pets NEED TO GET THIS WORKING

This function will be called with an array of objects. Each object represents an owner and will have a pets property, which will be an array of pet names. The function should return an array of **all** the pets' names.

If passed an empty array the function should return an empty array.

A typical array of owners is shown below:

[

{

name: 'Malcolm',

pets: ['Bear', 'Minu'],

},

{

name: 'Caroline',

pets: ['Basil', 'Hamish'],

},

];

Examples:

gatherPets([]); // should return []

gatherPets([{ name: 'Malcolm', pets: ['Bear', 'Minu'] }]);

// should return ['Bear', 'Minu']

gatherPets([

{ name: 'Malcolm', pets: ['Bear', 'Minu'] },

{ name: 'Caroline', pets: ['Basil', 'Hamish'] },

]);

// should return ['Bear', 'Minu', 'Basil', 'Hamish']

function gatherPets(people) {

//Define a new array to put all the results into

let newArray=[];

for (let i = 0; i <people.length; i++) {

if (people[i].pets !== '') {

newArray = newArray + " " +people[i].pets

}

}

console.log(newArray)

return (newArray)

}

gatherPets([]); // should return []

gatherPets([{ name: 'Malcolm', pets: ['Bear', 'Minu'] }]);

// should return ['Bear', 'Minu']

gatherPets([

{ name: 'Malcolm', pets: ['Bear', 'Minu'] },

{ name: 'Caroline', pets: ['Basil', 'Hamish'] },

]);

// should return ['Bear', 'Minu', 'Basil', 'Hamish']