JS Practice Problems

### Difficulty Level : Easy

### Create a function that takes two numbers as arguments and returns their sum.

### Write a function that takes an integer minutes and converts it to seconds.

1. Create a function that takes a number as an argument, increments the number by +1 and returns the result.
2. Create a function that takes the age in years and returns the age in days.
3. Create a function that takes voltage and current and returns the calculated **power**.
4. Write a function that returns the string "something" joined with a space " " and the given argument a.
5. Create a function that takes two arguments. Both arguments are integers, a and b. Return true if one of them is 10 or if their sum is 10.
6. Create a function that takes two strings as arguments and returns either true or false depending on whether the total number of characters in the first string is equal to the total number of characters in the second string.
7. Create a function that takes a name and returns a greeting in the form of a string. Don't use a normal function, use an **arrow function**.
8. Create a function that takes an array of 10 numbers (between 0 and 9) and returns a string of those numbers formatted as a phone number (e.g. **(555) 555-5555**).
9. Create a function that returns an array of strings sorted by length in **ascending** order.

Example:

sortByLength(["a", "ccc", "dddd", "bb"]) ➞ ["a", "bb", "ccc", "dddd"]

1. Create a function that takes an array of arrays with numbers. Return a new (single) array with the largest numbers of each.

Example:

* + - 1. findLargestNums([[4, 2, 7, 1], [20, 70, 40, 90], [1, 2, 0]]) ➞ [7, 90, 2]

1. Create a function that takes an array of numbers and returns the **second largest** number.

Example:

secondLargest([10, 40, 30, 20, 50]) ➞ 40

1. Create a function that takes an array of items, removes all duplicate items and returns a new array in the same sequential order as the old array (minus duplicates).

Example:

removeDups([1, 0, 1, 0]) ➞ [1, 0]

removeDups(["The", "big", "cat"]) ➞ ["The", "big", "cat"]

1. Create a function that takes an array of integers as an argument and returns a unique number from that array. All numbers except unique ones have *the same number of occurrences* in the array.

Example:

findSingleNumber([2, 2, 2, 3, 4, 4, 4]) ➞ 3

1. Create a function that takes two strings as arguments and returns the number of times the first string (the single character) is found in the second string.

Example:

charCount("c", "Chamber of secrets") ➞ 1

1. Create a function that takes a string and returns the number (count) of vowels contained within it.

Example:

countVowels("Celebration") ➞ 5

1. Given a string, create a function to reverse the case. All lower-cased letters should be upper-cased, and vice versa.

Example:

reverseCase("Happy Birthday") ➞ "hAPPY bIRTHDAY"

### Difficulty Level : Medium

1. Write a function that converts an object into an array, where each element represents a key-value pair in the form of an array.

Examples :

toArray({ a: 1, b: 2 }) ➞ [["a", 1], ["b", 2]]

toArray({ shrimp: 15, tots: 12 }) ➞ [["shrimp", 15], ["tots",12]]

toArray({}) ➞ []

1. Create a function that takes two numbers as arguments (num, length) and returns an array of multiples of num until the array length reaches length.

Examples :

arrayOfMultiples(7, 5) ➞ [7, 14, 21, 28, 35]

arrayOfMultiples(12, 10) ➞ [12, 24, 36, 48, 60, 72, 84, 96, 108, 120]

arrayOfMultiples(17, 6) ➞ [17, 34, 51, 68, 85, 102]

1. Create the function that takes an array with objects and returns the sum of people's budgets.

Examples :

getBudgets([

{ name: "John", age: 21, budget: 23000 },

{ name: "Steve", age: 32, budget: 40000 },

{ name: "Martin", age: 16, budget: 2700 }

]) ➞ 65700

getBudgets([

{ name: "John", age: 21, budget: 29000 },

{ name: "Steve", age: 32, budget: 32000 },

{ name: "Martin", age: 16, budget: 1600 }

]) ➞ 62600

1. Create a function that takes an array of objects like { name: "John", notes: [3, 5, 4]} and returns an array of objects like { name: "John", avgNote: 4 }. If a student has no notes (an empty array) then let's assume avgNote: 0.

Example :

[

{ name: "John", notes: [3, 5, 4]}

] ➞ [

{ name: "John", avgNote: 4 }

]

1. Create a function that moves all capital letters to the front of a word.

Examples :

capToFront("hApPy") ➞ "APhpy"

capToFront("moveMENT") ➞ "MENTmove"

capToFront("shOrtCAKE") ➞ "OCAKEshrt"

### Difficulty Level : Hard

1. Create a function that converts dash/underscore delimited words into camel casing. The first word within the output should be capitalized only if the original word was capitalized.

Examples :

toCamelCase("A-B-C") ➞ "ABC"

toCamelCase("the-stealth-warrior") ➞ "theStealthWarrior"

toCamelCase("The\_Stealth\_Warrior") ➞ "TheStealthWarrior"

1. Create a function that takes an array of strings and returns an array with only the strings that have numbers in them. If there are no strings containing numbers, return an empty array.

Examples :

numInStr(["1a", "a", "2b", "b"]) ➞ ["1a", "2b"]

numInStr(["abc", "abc10"]) ➞ ["abc10"]

numInStr(["abc", "ab10c", "a10bc", "bcd"]) ➞ ["ab10c", "a10bc"]

numInStr(["this is a test", "test1"]) ➞ ["test1"]

1. Write a function that takes a list of hours and returns the total weekly salary.

* The input list hours is listed sequentially, ordered from Monday to Sunday.
* A worker earns $10 an hour for the first 8 hours.
* For every overtime hour, he earns $15.
* On weekends, the employer pays double the usual rate, regardless how many hours were worked previously that week. For instance, 10 hours worked on a weekday would pay 80+30 = $110, but on a weekend it would pay 160+60 = $220.

Examples :

weeklySalary([8, 8, 8, 8, 8, 0, 0]) ➞ 400

weeklySalary([10, 10, 10, 0, 8, 0, 0]) ➞ 410

weeklySalary([0, 0, 0, 0, 0, 12, 0]) ➞ 280

1. Create a function which takes in an encoded string and returns an object according to the following example:

Examples :

parseCode("John000Doe000123") ➞ {

firstName: "John",

lastName: "Doe",

id: "123"

}

parseCode("michael0smith004331") ➞ {

firstName: "michael",

lastName: "smith",

id: "4331"

}

parseCode("Thomas00LEE0000043") ➞ {

firstName: "Thomas",

lastName: "LEE",

id: "43"

}

1. Create a function that takes the current day (e.g. "2022-09-30"), an array of date objects and returns the "current streak" (i.e. number of consecutive days in a row).

Examples :

currentStreak("2022-09-23", [

{"date": "2022-09-18"},

{"date": "2022-09-19"},

{"date": "2022-09-21"},

{"date": "2022-09-22"},

{"date": "2022-09-23"}]) ➞ 3

currentStreak("2022-09-25", [

{"date": "2022-09-16"},

{"date": "2022-09-17"},

{"date": "2022-09-21"},

{"date": "2022-09-22"},

{"date": "2022-09-23"}]) ➞ 0