Preguntas de Preparación Devsu Code Jam 2021

El examen consistirá de aproximadamente 3 preguntas fáciles, 3 medias y 2 difíciles, cada una con su propio puntaje. Debes resolver tanto como puedas en 3 horas.

En este documento detallamos preguntas de ejemplo, que servirán de preparación para el concurso.

1. Series

Look at this series: 7, 6, 8, 4, 9, 2, 10, 0, 11, -2, …

Create a function that receives two integers: x and y. If any of them are 0 or negative, or if they are greater than 255, the function should return -1

Otherwise, the function should return the sum of the x and y elements of the series.

For example: If the function receives x=1, y=3, it should return: 15. (Because the sum of the first plus the third argument is 7+8=15). If the function receives x=8, y=9, it should return 11. (Because the sum of the 8th plus the 9th element is 0+11=11).

The function will receive 2 integers, return an integer.

2. Prediction Search Engine

We are trying to implement a new prediction search engine, for the search engine to have a good performance it will not start predicting until the user types in the **second** character, the search engine should only return the first **3** predictions **ordered** in alphabetical order.

Given a set of elements, write a method that returns a `String[][]` containing all the predictions generated by the search engine, while the user typed the query. If the user typed a 7 letter word (ie. `cartoon`), the method should return an array containing 5 arrays, each of these arrays contains the predictions generated by adding each character to the query, first query: "ca", second query: "car", third query: "cart" and so on. If no results were found on a particular query, then that particular result will be an empty array.

There are no null inputs, nor strings that are of length less than 2.

Example:

Input

```
query: cartoons
```

```
elements: [car, cartouches, carpet, cartoonist, carrot, cared, carton,
captain, cartoon, carter]
```

Output

```
[
[captain, car, cared],
[car, cared, carpet],
[carter, carton, cartoon],
[carton, cartoon, cartoonist],
[cartoon, cartoonist],
[cartoon, cartoonist],
[]
]
```

3. Perfect numbers

A perfect number is a positive integer that is equal to the sum of its proper divisors. For example, 6 is a perfect number because 6=1+2+3.

Create a function that receives two values X and Y and return the smaller perfect number found, which is greater or equal than X and lower or equal than Y. If no perfect number found, it should return -1.

For example, if the function receives X=5, Y=7, it should return 6, because 6 is the smaller perfect number between 5 and 7.

The function will receive two integers and return one integer.

4. Sorting

For this problem, you are required to write a function that receives an array of numbers and should return the array sorted using for the comparison of their last digit, if their last digit is the same you should check the second to last and so on.

Example:

Input: `[1, 10, 20, 33, 13, 60, 92, 100, 21]`
Output: `[100, 10, 20, 60, 1, 21, 92, 13, 33]`

5. Eerie Mob

We all love emojis. ASCII emojis have existed for a while now, including this guy: **(-_-)**. You can represent a **mob** of these like this: `(-(-_(-_-(-_(-_(-_-)_-)_-)-_-)_-)-)`. **Looks scary!** That mob has 11 guys!

The rules to create a mob are as follows:

1. A mob can have **between 1 and 255 guys, 255 included**.
2. If there's an **even** number of guys, then there should be one more on the left than on the right: **eg 4:** `(-_(-_(-_-)_-)`
3. There are four different types of guys: a complete guy `(-_-)`, a side guy `_-)`, a partial guy `-_-)` and a final guy `-)`.
    a. The complete guy must be in the middle, and there must only be one per mob.
    b. A final guy must be on both sides, as long as there are **strictly more than 7 guys**.
    c. Let's say the **complete guy** is at the kth position. Every `(k ± 3n)`th guy must be a partial guy.
    d. All the rest are side guys.

An example, a mob that has 14 guys is this one.

```
(-(-_-(-_(-_(-_-(-_(-_(-_-)_-)_-)-_-)_-)_-)-)
```

Create a function that receives an `integer`, indicating how many guys are on your mob and returns a `string` with the resulting mob. If the parameter does not comply with the first rule, or if it's null or empty, then the function should return `(O_o)`

6. Product subarray

Given an **array of floating point numbers**, find the contiguous subarray with the largest product.

The subarray can be of any length, it could even be the whole array. The input will never be `null`.

Example:

Input: `[-3.2, 4.2, 7, 5.4, -2.2, -2.5]`
Output: `[-3.2, 4.2, 7, 5.4, -2.2]`

7. Toeplitz matrix

In linear algebra, a Toeplitz matrix is an **M x N** matrix in which all of the diagonals from top left to bottom right have the same value. You must make a function that takes an integer matrix as the input, and returns an integer, indicating how many distinct elements there are. If the matrix is **NOT** a Toeplitz matrix, return `-1`. Remember that single elements count as a diagonal. No input will be `null`.

Example:

Input:
```
1 2 3 4 8 1
5 1 2 3 4 8
4 5 1 2 3 4
7 4 5 1 2 3
```

Output: `7`