Intern Task 2 - JavaScript

A popular computer game that got some bad publicity from some priest (perhaps because developers decided to steal his cane and implement it into their work) contains runes that grant unimaginable powers. Almighty mage Ecmascriptus found that these runes can be melded into runic words. The power of each word is measured by the sum of the powers of its constituent runes minus the number of the runes used to construct the runic word:

$$P = r_1 + r_2 + \ldots + r_n - n$$

where P = power of runic word, r - rune, n - number of runes.

Example:

Runic word: Eth-El-Tir-Nef
Power of runic word = 31 + 28 + 9 + 7 - 4 = 71
Length of runic word = 4

After many attempts and mishaps Ecmascriptus came up which rune combinations can generate harmful side effects - explosions, disabilities and even death of the user. **(For example Eld-Lum-Sur would be incorrect because Sur can't be used when Eld is present in the runic word)**

Your task is to make the mage's work a little easier.

The first function should accept one argument - **runic word length** (in runes, not characters) and return an array of ten most powerful runic words with length equal to the provided length.

- The function should output an Array of 10 most powerful valid runic words, and their powers;
- Runic words must be sorted by their power (most powerful word goes first);
- Each rune can be used only once;
- Runes in runic word must be sorted by power (most powerful rune goes first);
- Runes in runic word should be joined with a dash (-);
- If there are not enough runes to create 10 runic words of given length, return as many as you can;
- Don't return any incomplete runic word.

For example the most powerful runic word for *length = 2* is **'Eld-Lum'** so it should be first in the returned Array.

The second function should tell the Mage if a word is correct and what is it's power. Create function that will accept runic word written in runic word format ("Eld-Lum"). That function will return runic word power if it's valid (according to the rules provided for the word generating function) otherwise return an error..

Task should be done in single js file named ***CCFrontTask2.js*** and should have exported function that generates runic words array (node.js style), and exported function that validates runic word and return runic word power or an error if it's invalid.

Function generating Runic Words

```
exports.generateRunicWords = length => {

        ....

        return runicWords;

}
```

Example output:

```
[

        {
                word: ‘Eld-Lum’,
                power: 63
        },
        …
]
```

Where length is a number and function generates Array with objects sorted by power:

Warning: Function should cover all input possibilities:

- **When input is incorrect return proper error message instead of array (depends of error type - eg. if input is empty then return something like "Input cannot be empty" etc.);**

Function validating and measuring Runic Words

```
exports.checkRunicWord = runicWord => {

        ....

        return runicWordObject;

}
```

Example output:

```
        63
```

Where runicWord is a string and function generates power of runic word

Warning: Function should cover all input possibilities:

- **When input is incorrect return proper error message instead of runic word power (depends of error type - eg. if input is empty then return something like "Input cannot be empty" etc.);**

What will be evaluated:

- Application should work properly, output must pass our tests suite
- Readability and quality of code:
    - Using good practices;
    - Using js methods;
    - Proper naming convention;
    - Writing comments in code when necessary;

Runic table:

Rune	Rune's power	Can't be linked with
El	28	Ort
Eld	33	Sur
Tir	9	Eth
Nef	7	Ist
Eth	31	Tir
Ith	22	Pul
Tal	8	Io
Ral	25	Um
Ort	18	El
Thul	13	Sol
Amn	6	Fal
Sol	10	Thul
Shael	17	Lem
Dol	11	Hel
Hel	12	Dol
Io	20	Tal
Lum	32	Gul
Ko	27	Mal
Fal	14	Amn
Lem	26	Shall
Pul	15	Ith
Um	16	Ral
Mal	21	Ko
Ist	4	Nef
Gul	23	Lum
Vex	24	Ohm
Ohm	1	Vex
Lo	2	Cham
Sur	30	Eld
Ber	3	
Jah	5	Zod
Cham	29	Lo
Zod	19	Jah