1. Escreva uma função que receba 2 parâmetros, uma String e um Inteiro. Essa função deve ter como retorno outra String.O objetivo da função é remover os caracteres repetidos consecutivamente e que extrapolam o limite (valor inteiro) passado como segundo parâmetro, ou seja, se o limite for 2, os caracteres podem ser repetidos no máximo duas vezes. Exemplos:
   1. function removeRepeated("xxxyzzaaayy", 1) => "xyzay"
   2. function removeRepeated("xxxyzzaaayy", 2) => "xxyzzaayy"

Considere utilizar o modelo de implementação abaixo:

|  |
| --- |
| */\*\* \* @param {string} string   \* @param {number} limit  \* @return {string}   \*/* const removeRepeated = (string, limit) => {     *//Your logic goes here*     return null; }  export default removeRepeated; |

1. Escreva uma função que receba 2 parâmetros, uma String e um Inteiro.Essa função deve ter como retorno outra String. O Objetivo da função é substituir o caracter "?" criando uma repetição do caracter que está antes dele o numero de vezes do parametro Inteiro. Caso o caracter "?" seja o primeiro da String fornecida ele deve repetir o último caracter da String. Exemplos:
   1. function replaceChar("x?yzza?y?", 1) => "xxyzzaayy"
   2. function replaceChar("x?yzza?y?", 2) => "xxxyzzaaayyy"
   3. function replaceChar("?x?yzza?y?", 2) => "yyxxxyzzaaayyy"

Considere utilizar o modelo de implementação abaixo:

|  |
| --- |
| */\*\* \* @param {string} string   \* @param {number} times  \* @return {string}   \*/* const replaceChar = (string, times) => {     *//Your logic goes here*     return null; }  export default replaceChar; |

1. Escreva uma função que receba 2 Strings.Essa função deve ter como retorno um número. O Objetivo da função é ordenar as 2 strings alfabeticamente e extrair de cada uma a maior e a menor letra. Trasforme essa letra em um número considerando que o Alfabeto é uma sequencia numérica e a letra "a" é igual a 1 a letra "b" é igual a 2 e assim sucessivamente. O retorno da função deve ser a soma da multiplicação entre a maior letra da primeira String e a  menor letra da segunda String com a multiplicação entre a maior letra da segunda String e a menor letra da primeira String.

retorno =(maiorLetraString1 menorLetraString2 ) + (maiorLetraString2 menorLetraString1)

* 1. function lettersSum("dacb", "gdab") =>  11
  2. function lettersSum("dcb", "gdab") =>  18

Considere utilizar o modelo de implementação abaixo:

|  |
| --- |
| */\*\* \* @param {string} string1  \* @param {string} string2 \* @return {number}   \*/* const lettersSum = (string1, string2) => {     *//Your logic goes here*     return null; }  export default lettersSum; |