1. Verificar entrada de texto

Escreva uma função em JavaScript para verificar se um campo de entrada ('input') é uma string ou não.

Dados de teste :

```
console.log(is_string('w3resource'));
true
console.log(is_string([1, 2, 4, 0]));
false
```

Clique aqui para ver a solução.

2. Verificar string em branco

Escreva uma função em JavaScript para verificar se uma string está vazia ou não.

Dados de teste :

```
console.log(is_Blank(''));
console.log(is_Blank('abc'));
true
false
```

Clique aqui para ver a solução.

3. Converter uma string em um array de palavras

Escreva uma função em JavaScript para dividir uma string e convertê-la em um array de palavras.

Dados de teste :

```
console.log(string_to_array("Robin Singh"));
["Robin", "Singh"]
```

Clique aqui para ver a solução.

4. Extrair caracteres

Escreva uma função em JavaScript para extrair um número específico de caracteres de uma string.

Dados de teste :

console.log(truncate_string("Robin Singh",4));
"Robi"

Clique aqui para ver a solução .

5. Abreviar o nome

Escreva uma função em JavaScript para converter uma string em sua forma abreviada.

Dados de teste :

console.log(abbrev_name("Robin Singh"));
"Robin S."

Clique aqui para ver a solução .

6. Ocultar endereço de e-mail

Escreva uma função em JavaScript que oculte endereços de e-mail para impedir o acesso não autorizado.

Dados de teste :

console.log(protect_email(" robin_singh@example.com "));
" robin...@example.com "

Clique aqui para ver a solução .

7. Parametrizar string

Escreva uma função em JavaScript para parametrizar uma string.

Dados de teste :

console.log(string_parameterize("Robin Singh dos EUA."));
"robin-singh-dos-eua"

Clique aqui para ver a solução .

8. Coloque a primeira letra em maiúscula.

Escreva uma função em JavaScript para colocar a primeira letra de uma string em maiúscula.

Dados de teste :

console.log(capitalize('exercícios de strings em JS'));
"Exercícios de strings em JS"

Clique aqui para ver a solução .

9. Escreva cada palavra em maiúscula.

Escreva uma função em JavaScript para colocar em maiúscula a primeira letra de cada palavra em uma string.

Dados de teste :

console.log(capitalize_Words('exercícios de strings em JS'));
"Exercícios de Strings em JS"

Clique aqui para ver a solução .

10. Trocar de capa

Escreva uma função em JavaScript que receba como parâmetro uma string contendo letras maiúsculas e minúsculas. A função deve converter letras maiúsculas em minúsculas e vice-versa.

Dados de teste :

console.log(swapcase('AaBbc'));
"aAbBC"

Clique aqui para ver a solução .

11. Camelizar string

Escreva uma função em JavaScript para converter uma string em camel case.

Dados de teste :

```
console.log(camelize("Exercícios de JavaScript"));
console.log(camelize("Exercícios de JavaScript"));
console.log(camelize("Exercícios de JavaScript"));
"Exercícios de JavaScript"
"Exercícios de JavaScript"
"Exercícios de JavaScript"
```

Clique aqui para ver a solução .

12. Descamelizar string

Escreva uma função em JavaScript para desembaralhar uma string.

Dados de teste :

```
console.log(uncamelize('helloWorld'));
console.log(uncamelize('helloWorld','-'));
console.log(uncamelize('helloWorld','_'));
"hello world"
"hello-world"
"hello_world"
```

Clique aqui para ver a solução .

13. Repetir sequência

Escreva uma função em JavaScript para concatenar uma string dada n vezes (o padrão é 1).

Dados de teste :

```
console.log(repetir('Ha!'));
console.log(repetir('Ha!',2));
console.log(repetir('Ha!',3));
"Ah!"
"Ha! Ha!"
"Ha! Ha! Ha!"
```

Clique aqui para ver a solução .

14. Inserir em String

Escreva uma função em JavaScript para inserir uma string dentro de outra string em uma posição específica (o padrão é 1).

Dados de teste :

console.log(insert('Estamos fazendo alguns exercícios.'));
console.log(insert('Estamos fazendo alguns exercícios.','JavaScript '));
console.log(insert('Estamos fazendo alguns exercícios.','JavaScript ',18));
"Estamos fazendo alguns exercícios."
"JavaScript Estamos fazendo alguns exercícios."
"Estamos fazendo alguns exercícios de JavaScript."

Clique aqui para ver a solução .

15. Formato Humanizado

Escreva uma função em JavaScript que formate um número em uma string legível para humanos com o sufixo correto, como 1º, 2º, 3º, etc.

Dados de teste :

console.log(humanize_format());
console.log(humanize_format(1));
console.log(humanize_format(8));
console.log(humanize_format(301));
console.log(humanize_format(402));
"1º"
"8º"
"301º"
"402º"

Clique aqui para ver a solução .

16. Truncar string com reticências

Escreva uma função em JavaScript para truncar uma string se ela for maior que o número de caracteres especificado. As strings truncadas terminarão com uma sequência de reticências traduzíveis ("...") (por padrão) ou com os caracteres especificados.

Dados de teste :

```
console.log(text_truncate('Estamos fazendo exercícios de strings em JS.'))
console.log(text_truncate('Estamos fazendo exercícios de strings em JS.',19))
console.log(text_truncate('Estamos fazendo exercícios de strings em JS.',15,'!!'))
"Estamos fazendo exercícios de strings em JS."
"Estamos fazendo JS ..."
"Estamos fazendo !!"
```

Clique aqui para ver a solução .

17. Corte o barbante em pedaços.

Escreva uma função em JavaScript para dividir uma string em partes de um determinado comprimento.

Dados de teste :

```
console.log(string_chop('w3resource'));
console.log(string_chop('w3resource',2));
console.log(string_chop('w3resource',3));
["w3resource"]
["w3", "re", "so", "ur", "ce"]
["w3r", "eso", "urc", "e"]
```

Clique aqui para ver a solução .

18. Contar ocorrências de subcadeias

Escreva uma função em JavaScript para contar as substrings em uma string.

Dados de teste :

```
console.log(count("A raposa marrom veloz pula sobre o cachorro preguiçoso", 'a'));
Saída:
2
console.log(count("A raposa marrom veloz pula sobre o cachorro preguiçoso",
'raposa',false));
Saída:
1
```

Clique aqui para ver a solução .

19. Representação Binária Inversa

Escreva uma função em JavaScript que receba um número inteiro positivo e inverta a representação binária desse número. Por fim, retorne a versão decimal da string binária.

Dados de teste :

(100) -> 19
Explicação:
A representação binária de 100 é 1100100.
O inverso de 1100100 é 10011.
A forma decimal de 10011 é 19.
(1134) -> 945
Explicação:
A representação binária de 1134 é 10001101110.
O inverso de 10001101110 é 1110110001.
A forma decimal de 1110110001 é 945.

Clique aqui para ver a solução .

20. Ajuste o comprimento da corda ao nível desejado.

Escreva uma função em JavaScript que preencha (à esquerda, à direita) uma string até atingir um comprimento específico.

Dados de teste :

console.log(formatted_string('0000',123,'l'));
console.log(formatted_string('00000000',123,''));
Saída:
"0123"
"12300000"

Clique aqui para ver a solução .

21. Repita a sequência várias vezes

Escreva uma função em JavaScript para repetir uma string por um período de tempo especificado.

Dados de teste :

```
console.log(repeat_string('a', 4));
console.log(repeat_string('a'));
```
Saída:
"aaaa"
"Erro na string ou na contagem."

Clique aqui para ver a solução .

22. Substring após o caractere

Escreva uma função em JavaScript para obter uma parte de uma string após um caractere específico.

Dados de teste :

```
console.log(subStrAfterChars('w3resource: Exercícios de JavaScript', ':','a'));
console.log(subStrAfterChars('w3resource: Exercícios de JavaScript', 'E','b'));
```
Saída:
"w3resource"
"exercícios"

Clique aqui para ver a solução .

23. Aparar espaços

Escreva uma função em JavaScript para remover espaços em branco no início e no final de uma string.

Dados de teste :

```
console.log(strip('w3resource '));
console.log(strip(' w3resource'));
console.log(strip(' w3resource '));
```
Saída:
"w3resource"
"w3resource"
"w3resource"

Clique aqui para ver a solução .

24. Truncar por palavras

Escreva uma função em JavaScript para truncar uma string para um determinado número de palavras.

Dados de teste :

console.log(truncate('A raposa marrom veloz pula sobre o cachorro preguiçoso', 4));
Saída:
"A raposa marrom veloz"

Clique aqui para ver a solução .

25. Ordenar a string alfabeticamente

Escreva uma função em JavaScript para ordenar alfabeticamente uma determinada string.

Alfabetizar string: Uma string individual pode ser alfabetizada. Isso reorganiza as letras para que sejam classificadas de A a Z.
Dados de teste :
console.log(alphabetize_string('United States'));
Saída:
"SUadeeinsttt"

Clique aqui para ver a solução .

26. Remover a primeira ocorrência

Escreva uma função em JavaScript para remover a primeira ocorrência de uma determinada 'string de busca' de uma string.

Dados de teste :

console.log(remove_first_occurrence("The quick brown fox jumps over the lazy dog", 'the'));
Saída:
"The quick brown fox jumps over lazy dog"

Clique aqui para ver a solução .

27. ASCII para Hexadecimal

Escreva uma função em JavaScript para converter o formato ASCII para hexadecimal.

Dados de teste :

```
console.log(ascii_to_hexa('12'));
console.log(ascii_to_hexa('100'));
```

Saída:
"3132"
"313030"

Clique aqui para ver a solução .

28. Hexadecimal para ASCII

Escreva uma função em JavaScript para converter o formato hexadecimal para ASCII.

Dados de teste :

```
console.log(hex_to_ascii('3132'));
console.log(hex_to_ascii('313030'));
```

Saída:
"12"
"100"

Clique aqui para ver a solução .

29. Encontrar palavra em uma sequência de caracteres

Escreva uma função em JavaScript para encontrar uma palavra dentro de uma string.

Dados de teste :

```
console.log(search_word('The quick brown fox', 'fox'));
console.log(search_word('aa, bb, cc, dd, aa', 'aa'));
```

Saída:
"'fox' foi encontrado 1 vez."
"'aa' foi encontrado 2 vezes."

Clique aqui para ver a solução .

30. A corda termina com

Escreva uma função em JavaScript que verifique se uma string termina com um sufixo específico.

Dados de teste :

```
console.log(string_endsWith('JS PHP PYTHON','PYTHON'));
true
console.log(string_endsWith('JS PHP PYTHON',''));
false
```

Clique aqui para ver a solução .

31. Caracteres de escape HTML

Escreva uma função em JavaScript para escapar caracteres especiais (&, <, >, ', ") para uso em HTML.

Dados de teste :

```
console.log(escape_html('PHP & MySQL'));
"PHP & MySQL"
console.log(escape_html('3 > 2'));
"3 > 2"
```

Clique aqui para ver a solução .

32. Remover caracteres ASCII não imprimíveis

Escreva uma função em JavaScript para remover caracteres ASCII não imprimíveis .

Dados de teste :

```
console.log(remove_non_ascii('???????PHP-MySQL??????'));
"PHP-MySQL"
```

Clique aqui para ver a solução .

33. Remover caracteres que não fazem parte de palavras

Escreva uma função em JavaScript para remover caracteres que não sejam palavras.

Dados de teste :

```
console.log(remove_non_word('PHP ~!@#$%^&*()+`-={}[]|\\:";\'/?><., MySQL'));
"PHP - MySQL"
```

Clique aqui para ver a solução .

34. Converter para Maiúsculas e Minúsculas

Escreva uma função em JavaScript para converter uma string para o formato de título (primeira letra maiúscula).

Dados de teste :

console.log(sentenceCase('Exercícios de PHP. Exercícios de Python.'));
"Exercícios de PHP. Exercícios de Python."

Clique aqui para ver a solução .

35. Remover tags HTML/XML

Escreva uma função em JavaScript para remover tags HTML/XML de uma string.

Dados de teste :

console.log(strip_html_tags(' Exercícios de PHP '));
"Exercícios de PHP"

Clique aqui para ver a solução .

36. Número com Preenchimento de Zeros

Escreva uma função em JavaScript para criar um valor preenchido com zeros, com a opção de adicionar ou remover os sinais + e -.

Dados de teste :

console.log(zeroFill(120, 5, '-'));
"+00120"
console.log(zeroFill(29, 4));
"0029"

Clique aqui para ver a solução .

37. Comparação que ignora maiúsculas e minúsculas

Escreva uma função em JavaScript para testar a comparação de strings sem distinção entre maiúsculas e minúsculas (exceto caracteres Unicode especiais).

Dados de teste :

```
console.log(compare_strings('abcd', 'AbcD'));
true
console.log(compare_strings('ABCD', 'Abce'));
false
```

Clique aqui para ver a solução .

38. Pesquisa que ignora maiúsculas e minúsculas

Escreva uma função em JavaScript para criar uma pesquisa que ignore maiúsculas e minúsculas.

Dados de teste :

```
console.log(case_insensitive_search('JavaScript Exercises', 'exercises'));
"Correspondência encontrada"
console.log(case_insensitive_search('JavaScript Exercises', 'Exercises'));
"Correspondência encontrada"
console.log(case_insensitive_search('JavaScript Exercises', 'Exercisess'));
"Não encontrado"
```

Clique aqui para ver a solução .

39. Retire a primeira letra maiúscula.

Escreva uma função em JavaScript para remover a primeira letra maiúscula de uma string.

Dados de teste :

```
console.log(Uncapitalize('Exercícios de strings em JS'));
"exercícios de strings em JS"
```

Clique aqui para ver a solução .

40. Retire a maiúscula de cada palavra.

Escreva uma função em JavaScript para remover a maiúscula da primeira letra de cada palavra de uma string.

Dados de teste :

console.log(unCapitalize_Words('Exercícios de Strings em JavaScript'));
"exercícios de strings em JavaScript"

Clique aqui para ver a solução .

41. Use maiúsculas em todas as palavras.

Escreva uma função em JavaScript para converter cada palavra da string para maiúscula.

Dados de teste :

console.log(capitalizeWords('exercícios de strings em JS'));
"EXERCÍCIOS DE STRINGS EM JS"

Clique aqui para ver a solução .

42. Elimine todas as letras maiúsculas do alfabeto.

Escreva uma função em JavaScript para converter cada palavra da string para minúsculas.

Dados de teste :

console.log(unCapitalizeWords('EXERCÍCIOS DE STRING EM JS'));
"exercícios de string em js"

Clique aqui para ver a solução .

43. O caractere é maiúsculo?

Escreva uma função em JavaScript para verificar se o caractere no índice fornecido é maiúsculo.

Dados de teste :

console.log(isUpperCaseAt('Exercícios de Strings em JavaScript', 1));
false

Clique aqui para ver a solução .

44. O caractere é minúsculo?

Escreva uma função em JavaScript para verificar se o caractere no índice (character) fornecido é minúsculo.

Dados de teste :

console.log(isLowerCaseAt('Exercícios de Strings em JavaScript', 1));
true

Clique aqui para ver a solução .

45. Sufixo numérico humanizado

Escreva uma função em JavaScript para obter um número escrito de forma legível, com o sufixo correto, como 1º, 2º, 3º ou 4º.

Dados de teste :

console.log(humanize(1));
console.log(humanize(20));
console.log(humanize(302));
"1º"
"20º"
"302º"

Clique aqui para ver a solução .

46. Começa com substring

Escreva uma função em JavaScript para verificar se uma string começa com uma string especificada.

Dados de teste :

console.log(startsWith('exercícios de string js', 'js'));
true

Clique aqui para ver a solução .

47. Termina com substring

Escreva uma função em JavaScript para verificar se uma string termina com uma string especificada.

Dados de teste :

console.log(endsWith('Exercícios de string JS', 'exercícios'));
true

Clique aqui para ver a solução .

48. Sucessor de Cadeia

Escreva uma função em JavaScript para obter o sucessor de uma string.

Observação: o sucessor é calculado incrementando os caracteres a partir do caractere alfanumérico mais à direita (ou do caractere mais à direita, caso não haja nenhum alfanumérico) na string. Incrementar um dígito sempre resulta em outro dígito, e incrementar uma letra resulta em outra letra com a mesma capitalização. Se o incremento gerar um "vai um", o caractere à esquerda dele é incrementado. Esse processo se repete até que não haja mais "vai um", adicionando um caractere adicional, se necessário.

Exemplo :

string.successor("abcd") == "abce"
string.successor("THX1138") == "THX1139"
string.successor("< >") == "< >"
string.successor("1999zzz") == "2000aaa"
string.successor("ZZZ9999") == "AAAA0000"

Dados de teste :

console.log(successor('abcd'));
console.log(successor('3456'));
"abce"
"3457"

Clique aqui para ver a solução .

49. Gerar GUID

Escreva uma função em JavaScript para obter um GUID (acrônimo para 'Identificador Globalmente Único'?) com o comprimento especificado, ou 32 por padrão.

Dados de teste :

console.log(guid());
console.log(guid(15));
"hRYilcoV7ajokxsYFl1dba41AyE0rUQR"
"b7pwBqrZwqaDrex"

Clique aqui para ver a solução .

50. Verifique se o palíndromo alfanumérico é válido.

Escreva um programa em JavaScript para verificar se uma determinada string contém caracteres alfanuméricos que são palíndromos, independentemente de caracteres especiais e maiúsculas/minúsculas.

Um palíndromo é uma palavra, número, frase ou outra sequência de símbolos que se lê da mesma forma de trás para frente, como as palavras "madam" ou "racecar", os registros de data e hora "11/11/11 11:11" e "02/02/2020" e a frase: "A man, a plan, a canal - Panama". A palavra finlandesa de 19 letras "saippuakivikauppias" (um vendedor de pedra-sabão) é o palíndromo de palavra única mais longo em uso cotidiano, enquanto o termo de 12 letras "tattarrattat" (de James Joyce em "Ulisses") é o mais longo em inglês.

Dados de teste :

('$22_|1372^2731|_22') -> verdadeiro
('12%^&2') -> falso
('234%$$%432') -> verdadeiro
(1234) -> "Deve ser uma string"
('aba%$aba') -> verdadeiro
('Aba%$aba') -> verdadeiro

Clique aqui para ver a solução .

51. Busca de strings de Boyer-Moore

Escreva uma função em JavaScript para implementar o algoritmo de busca de strings de Boyer-Moore.

Da Wikipédia,

Em ciência da computação, o algoritmo de busca de strings de Boyer-Moore é um algoritmo eficiente que serve como referência padrão na literatura prática sobre busca de strings. Foi desenvolvido por Robert S. Boyer e J. Strother Moore em 1977. O artigo original continha tabelas estáticas para calcular os deslocamentos de padrão, sem explicar como gerá-las. O algoritmo para gerar as tabelas foi publicado em um artigo subsequente; este artigo continha erros que foram posteriormente corrigidos por Wojciech Rytter em 1980.

O algoritmo pré-processa a sequência de caracteres que está sendo procurada (o padrão), mas não a sequência de caracteres na qual a busca está sendo feita (o texto). Portanto, ele é adequado para aplicações em que o padrão é muito mais curto que o texto ou onde ele persiste em múltiplas buscas. O algoritmo de Boyer-Moore utiliza informações coletadas durante a etapa de pré-processamento para pular seções do texto, resultando em um fator constante menor do que muitos outros algoritmos de busca de strings. Em geral, o algoritmo é executado mais rapidamente à medida que o comprimento do padrão aumenta. As principais características do algoritmo são a busca na cauda do padrão em vez do início e a busca ao longo do texto em saltos de múltiplos caracteres, em vez de pesquisar cada caractere individualmente.

O algoritmo de busca de strings de Boyer-Moore permite tempo de busca linear, pulando índices ao procurar um padrão dentro de uma string.

Dados de teste :

('ESTE É UM TEXTO DE TESTE', 'TESTE') -> 10
('ESTE É UM TEXTO DE TESTE', 'TESTE') -> 14

Clique aqui para ver a solução .

52. Exceder a verificação de palavras

Escreva um programa em JavaScript para determinar se uma determinada palavra excede o limite de palavras ou não.

Em palavras com mais de um caractere, há um espaço crescente entre dois caracteres adjacentes. Em ASCII, esse espaço representa a distância entre dois caracteres.

Dados de teste :

'acgl' -> verdadeiro
'aebc' -> falso

Clique aqui para ver a solução .

53. Verifique o estojo plano

Escreva uma função em JavaScript para verificar se uma determinada string está em formato FlatCase ou não.

Flat case: Como o nome indica, flat case refere-se ao uso de letras minúsculas, sem espaços entre as palavras.

Dados de teste :

('j') -> verdadeiro
('exercícios de java') -> falso
('Exercícios de JavaScript') -> falso
('exercícios de javascript') -> verdadeiro
(12356) -> "Deve ser uma string."

Clique aqui para ver a solução .

54. Verifique a caixa do kebab

Escreva uma função em JavaScript para verificar se uma determinada string está no formato Kebab-Case ou não.

Caso Kebab: "a raposa marrom rápida pula sobre o cachorro preguiçoso"

Semelhante ao snake case acima, exceto que hífens são usados em vez de sublinhados para substituir espaços. Também é conhecido como spinal case, param case, Lisp case em referência à linguagem de programação Lisp ou dash case (ou, ilustrativamente, como kebab-case).

Dados de teste :

('j') -> verdadeiro
('exercícios de java') -> falso
('Exercícios de JavaScript') -> falso
('exercícios de javascript') -> verdadeiro
(12356) -> "Deve ser uma string."

Clique aqui para ver a solução .

55. Verificação de pangramas

Escreva uma função em JavaScript para verificar se uma string é um pangrama ou não.

Um pangrama ou frase holoalfabética é uma frase que utiliza todas as letras de um determinado alfabeto pelo menos uma vez. Os pangramas têm sido usados para exibir tipos de letra, testar equipamentos e desenvolver habilidades em escrita à mão, caligrafia e digitação.

Dados de teste :

("A raposa marrom veloz pula sobre o cachorro preguiçoso") -> verdadeiro
("Valsa, ninfa má, para jigs rápidos irritam.") -> verdadeiro
("Os cinco magos do boxe saltam rapidamente.") -> verdadeiro
("Os magos do boxe saltam rapidamente.") -> falso
(12356) -> "Deve ser uma corda."

Clique aqui para ver a solução .

56. Verifique o caso Pascal

Escreva uma função em JavaScript para verificar se uma string está em Pascal case ou não.

A convenção de nomenclatura PascalCase coloca em maiúscula a primeira letra de cada palavra composta em uma variável. É uma boa prática no desenvolvimento de software usar nomes de variáveis descritivos.

Dados de teste :

("XmlStream") -> true
("IOStream") -> true
("javascript") -> false
(12356) -> "Deve ser uma string."

Clique aqui para ver a solução .

57. Reorganize para formar um palíndromo.

Escreva uma função em JavaScript que receba uma string e determine se ela pode ou não ser reorganizada para se tornar um palíndromo.

Quando uma palavra, frase ou sequência pode ser lida tanto de trás para frente quanto de frente para trás, é considerada um palíndromo. Exemplo: "madam" ou "nurses run".

Dados de teste :

("maamd") -> true
("civic") -> true
("IO") -> false
(12321) -> "Deve ser uma string."

Clique aqui para ver a solução .

58. Personagem mais frequente

Escreva um programa em JavaScript para encontrar o caractere mais frequente em uma determinada string.

Dados de teste :

("Senhora") -> "a"
("cívico") -> "c"
("OláL223LLL") -> "L"
(12321) -> "Deve ser uma string."

Clique aqui para ver a solução .

59. Palavra mais frequente

Escreva um programa em JavaScript para encontrar a palavra mais frequente em uma determinada string.

Dados de teste :

("A raposa marrom veloz pula sobre o cachorro preguiçoso") -> "a"
("Python é uma linguagem de programação de alto nível e propósito geral.") -> "python"
("Era o mesmo homem, ela tinha certeza. É sempre o mesmo, Chauncey.") -> "era"
(12321) -> "Deve ser uma string."

Clique aqui para ver a solução .

60. Inverter palavras

Escreva uma função em JavaScript para inverter as palavras em uma determinada string.

Dados de teste :

("abc") -> "cba"
("Exercícios de JavaScript") -> "tpircSavaJ sesicrexE"
(1234) -> "Deve ser uma string."

Clique aqui para ver a solução .

61. Subsequência Comum Mais Longa

Escreva uma função em JavaScript para encontrar o comprimento da maior subsequência presente entre duas sequências.

É importante entender que uma subsequência é uma sequência que aparece em uma ordem relativa semelhante, mas não é necessariamente contígua.

Dados de teste :

("abcda", "abcdef") -> 4
("ABCD", "ACBAD") -> 3
("pqr", "pqr") -> 3
("pqr", "abc") -> 0

Clique aqui para ver a solução .

62. Parênteses Válidos Mais Longos

Escreva uma função em JavaScript para obter o comprimento do maior parêntese válido (bem formado) em uma string contendo apenas os caracteres '[' e ']'.

Dados de teste :

A maior substring válida entre parênteses é "[]".
("[[]") -> 2
A maior substring válida entre parênteses é "[][]".
("][][]") -> 4
Nenhuma substring válida entre parênteses.
("") -> 0

63. Subsequência Palindrômica Mais Longa

Escreva uma função em JavaScript para encontrar o comprimento da maior subsequência palindrômica em uma determinada string.

Subsequências são sequências que podem ser criadas excluindo alguns ou todos os elementos de outra sequência sem alterar sua ordem.

Dados de teste :

("aaaba") -> 4
("maadaem") -> 5
("zkksk") -> 3
("ab") -> 1
("") -> ""