

Lista 04


Para cada um dos problemas a seguir, **encontre uma fórmula recursiva** apropriada, que contemple **caso base** e **passo indutivo**. Opte pela notação de funções. A seguir, implemente sua formulação recursiva na linguagem Javascript.


 Q1. N-ésimo termo da sequência $\{3, 6, 12, 24, 48, \dots\}$.

 Q2. N-ésimo termo da sequência $\{0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \dots\}$.

 Q3. Soma dos n primeiros números naturais.

 Q4. Fatorial de um número natural qualquer.

 Q5. Potência natural do número 2: 2^n . Naturalmente, você não deve utilizar o operador de expoente da linguagem.

 Q6. Resto da divisão entre dois números inteiros positivos fornecidos, n e m . Naturalmente, você não deve utilizar operadores de divisão da linguagem.

Ex: resto da divisão entre o número 31 e o 7.

$$31 - 7 = 24$$

$$24 - 7 = 17$$

$$17 - 7 = 10$$

$$10 - 7 = 3$$

$$3 < 7 \implies 3$$


[Formulação recursiva]


$$\text{resto}(n, m) = n, \forall n < m$$


$$\text{resto}(n, m) = \text{resto}(n - m, m), \forall n \geq m$$

 Q7. Quociente da divisão entre dois números inteiros positivos fornecidos, n e m . Naturalmente, você não deve utilizar operadores de divisão da linguagem.

 Q8. Máximo Divisor Comum (MDC) entre dois números inteiros positivos, n e m . Naturalmente, você não deve utilizar operadores de divisão da linguagem.


 Q9. Mínimo Múltiplo Comum (MMC) entre dois inteiros positivos fornecidos, n e m . Naturalmente, você não deve utilizar operadores de divisão da linguagem.

 Q10. Escreva uma função recursiva chamada `tamanho` que retorne o comprimento de uma string.

 **Q11.** Dado um número, devolva a soma total desse número multiplicado por cada número entre 1 e 10. Nome da função deve ser `multiSoma` .

 **Q12.** Escreva uma função recursiva que inverte uma string. Nome deve ser `inverte` .

 **Q13.** Criar uma função que calcula a soma dos quadrados diferentes até um número n. Nome deve ser `somaQuadrados` .

 **Q14.** Criar uma função recursiva chamada `repita` que pega dois parâmetros e repete a string n vezes. O primeiro parâmetro `txt` é a string a ser repetida e o segundo parâmetro é o número de vezes que a string deve ser repetida.

 **Q15.** Um vendedor tem uma série de cidades para visitar. Deve-se calcular o número total de caminhos possíveis a percorrer, visitando cada cidade uma vez antes de regressar à casa. Devolver o número total de caminhos possíveis que um vendedor pode percorrer, dadas n cidades. Nome deve ser `caminhos` .

Para cidades A, B e C, os caminhos possíveis seriam:

A -> B -> C


A -> C -> B


B -> A -> C

B -> C -> A

C -> B -> A

C -> A -> B

 **Q16.** Crie uma função chamada `dec2` que pega um número inteiro positivo na base 10 e o converte para uma outra base passada como argumento: base 2, base 8, ou base 16. A função deve retornar a string representativa do número nessa nova base.

 **Q17.** "A Conjectura de Collatz". Considere a seguinte operação sobre um número inteiro positivo arbitrário:
Se n é par -> $n / 2$. Se n é ímpar -> $n * 3 + 1$. Criar uma função chamada `collatz` para avaliar repetidamente estas operações, até atingir 1. Devolver o número de passos realizados. Ver o exemplo seguinte, usando 10 como entrada, com 6 passos:

1. 10 é par - $10 / 2 = 5$


2. 5 é ímpar - $5 * 3 + 1 = 16$


3. 16 é par - $16 / 2 = 8$


4. 8 é par - $8 / 2 = 4$

5. 4 é par - $4 / 2 = 2$

6. 2 é par - $2 / 2 = 1$ -> Atingiu 1, portanto, retorna 6 (passos).

 **Q18.** Função que determina se uma string é um `palíndromo` . Você deve desconsiderar todos caracteres que não sejam letras. Tanto faz maiúsculas e minúsculas. Você deve desconsiderar acentuações (substituir pelas letras sem acentos). A chamada `palindromo("Socorram-me, subi no ônibus em marrocos!")`, por exemplo, deve retornar `TRUE`.

 **Q19.** Função para retornar o número de dígitos do valor total de permutações possíveis para o conjunto formado por n elementos diferentes. O valor de n é a entrada da função. Ex: `ndigitospermut(5)` ---> 3 porque 5 elementos diferentes permitem 120 permutações e como 120 possui três dígitos.... resultado = 3.

-  **Q20.** A operação de deslocamento de bit à direita (shift right) é semelhante ao piso (*floor*) da divisão de inteiros por potências de 2. Portanto, o processo é repetitivo e pode ser feito de forma recorrente. Existe um operador para realizar essa operação de deslocamento em JS; trata-se do operador lógico `>>`. Você deve criar uma função que imita esse operador, sem usar o operador, obviamente! Ex: `deslocaDir(80,3) ---> 10` pois $80/2^3 = 10$. Veja que deslocamento à direita significa deslocar 3 bits para a direita na representação binária do número. Ou seja, se $80_2 = 1010000$, ao deslocarmos 3 bits para direita, temos `0001010`, que corresponde ao número 10 em binário. Voilà!
-