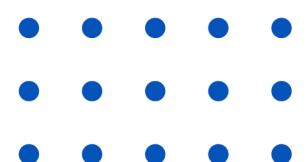
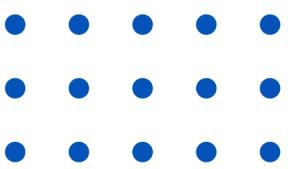


Strings e Arrays

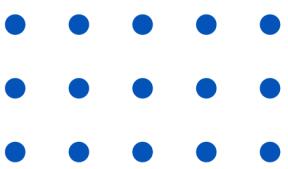




Declaração de Strings



- Como vimos anteriormente, Strings são os tipos referentes à **textos**
- Temos 3 maneiras de escrever uma string:
 - Aspas Duplas: "Olá Mundo"
 - Aspas Simples: 'Olá Mundo'
 - Crase (Template String ou Template Literals):
`Olá Mundo`



Concatenação de Strings

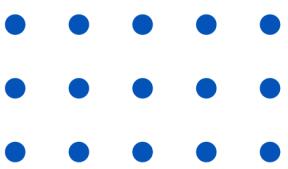


- Também podemos juntar várias strings para formar uma nova
- Chamamos esse processo de **concatenação** e utilizamos o sinal de **+** para fazê-lo

```
const nome = "Mika"  
const idade = 27  
const frase = "Meu nome é " + nome + " e tenho " + idade + " anos"
```



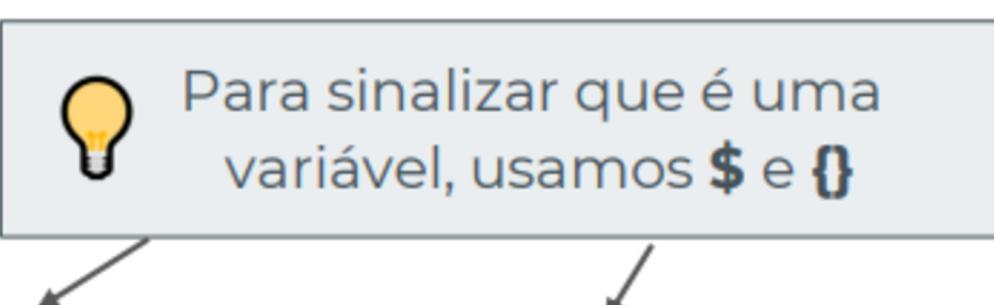
É necessário colocar o espaço para separar palavras

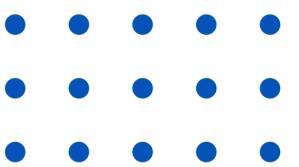


Template Strings

- Não há diferença entre usar aspas simples ou duplas!
- A única diferente é a **Template String**, pois ela nos permite colocar variáveis javascript no meio da string

```
const nome = "Mika"  
const idade = 27  
const frase = `eu nome é ${nome} e tenho ${idade} anos`  
// Meu nome é Mika e tenho 27 anos
```

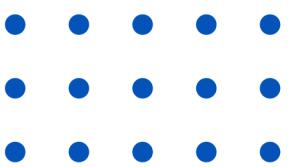




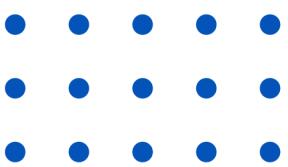
Crie um programa com variáveis do seu nome e sua **COR** favorita, seu **FILME** favorito, seu **TIME** do coração e imprima a mensagem:

Meu nome é **NOME** e minha cor favorita é **COR**. Gosto muito de estudar no SENAC e curto o filme **FILME**. Além disso, meu time é o **TIME**.

Faça o exercício duas vezes, utilizando concatenação e template strings.



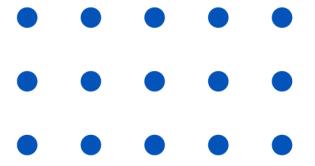
Protótipo de Strings



Propriedade length

- A propriedade **length** nos diz qual é o **tamanho** de uma string, incluindo espaços

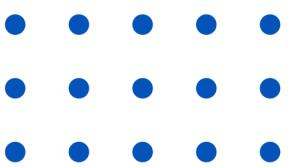
```
const nome = "Vitor Hugo"  
  
console.log(nome.length) // 10
```



Método `toLowerCase()`

- o método `toLowerCase()` transforma todas as letras da sua string em minúsculas

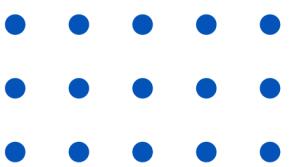
```
const frase = "OieEeEee!"  
const fraseMinuscula = frase.toLowerCase()  
// fraseMinuscula = oieeeee!
```



Método `toUpperCase()`

- o método `toUpperCase()` transforma todas as letras da sua string em maiúsculas

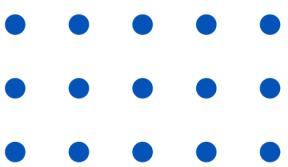
```
const frase = "OieEeEee!"  
const fraseMaiuscula = frase.toUpperCase()  
// fraseMaiuscula = OIEEEEEEE!
```



Método trim()

- O método **trim()** retira os espaços que existem antes e depois da sua string
- Útil em formulários como por exemplo de login!

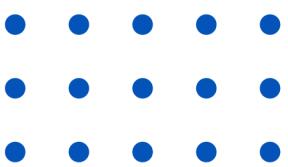
```
const email = "  mik@gmail.com  "
console.log(email.trim())
// "mik@gmail.com"
```



Método `includes(caracteres)`

- O método `includes(caracteres)` determina se um conjunto de caracteres pode ser encontrado dentro de outra string, retornando **true** ou **false**

```
const frase = "Hoje comi cenoura"  
frase.includes("cenoura") // true  
frase.includes("batata") // false
```



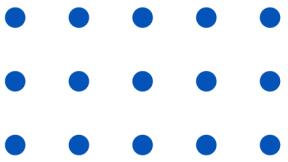
Método `replaceAll(chars1, chars2)`

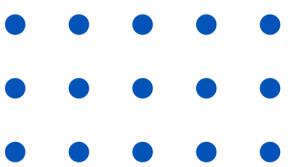


- O método `replaceAll(chars1, chars2)` troca todas as ocorrências de um conjunto de caracteres (`chars1`) por alguma outra coisa (`chars2`)

```
const frase = "Hoje comi cenoura, adoro cenoura"  
const novaFrase = frase.replaceAll("cenoura", "batata")  
// novaFrase = Hoje comi batata, adoro batata
```

Arrays





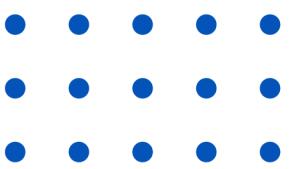
O que são arrays?



- Arrays nada mais são do que **listas de elementos**
 - **Ex:** lista de compras, lista de alunos, lista de números da loteria, lista telefônica...
- No javascript, usamos colchetes para agrupar os itens de uma lista:

```
const listaDeCompras = ["batata", "alface", "queijo"]
```

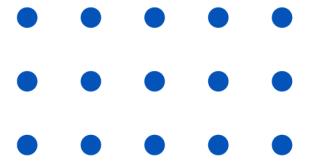
```
const listaDeNumerosMega = [2, 13, 26, 35, 41, 60]
```



O que são arrays?

- Podemos colocar elementos de **qualquer tipo** que vimos até agora dentro de um array!
 - Números, strings e booleanos
- Também podemos ter elementos de tipos diferentes dentro de um mesmo array

```
const meuArray = ["banana", 15, true]
```

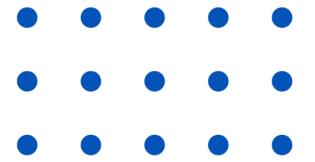


O que são arrays?



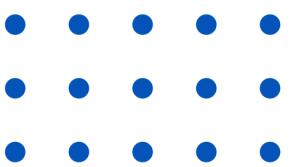
VALORES

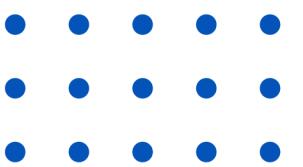


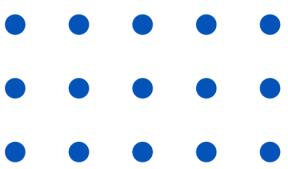


O que são arrays?









Acessando um elemento



- Em um array, acessamos os elementos através da **posição**(índice) deles na lista!
- Funciona como se fosse uma lista numerada:

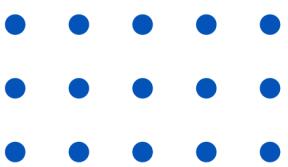
Listade Compras

1. Abacate
2. Banana
3. Tomate



Qual é o **item na posição 2?**

Resposta: Banana



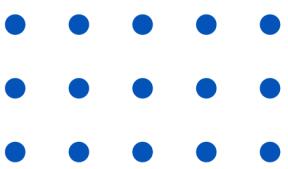
Acessando um elemento



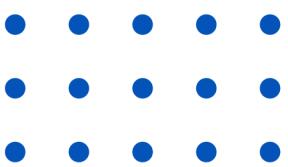
- Mas no caso dos arrays, a numeração não começa no 1, **mas sim no 0!**
- Para acessar um item, colocamos a sua posição (**índice**) entre colchetes após o nome do array

Lista de Compras	
0.	Abacate
1.	Banana
2.	Tomate

```
const listaDeCompras = ["Abacate", "Banana", "Tomate"]
const segundoItem = listaDeCompras[2] // "Tomate"
```



- Crie um array com pelo menos 5 raças de cachorro
- Imprima no console a raça correspondente a primeira posição e uma que você goste.



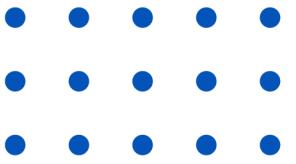
EX2.

Você vai montar um guarda-roupa virtual usando JavaScript. Crie um array com peças de roupa disponíveis e, a partir dele, selecione combinações específicas para diferentes ocasiões.

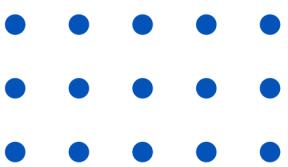
Situações para montar os looks:

1. Look para um passeio no parque
2. Look para uma entrevista de emprego
3. Look para um show à noite
4. Look para ficar em casa assistindo filmes





Protótipo de Arrays

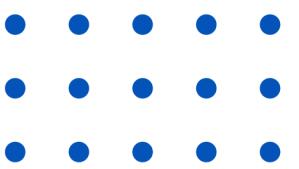


Propriedade length



- A propriedade **length** nos diz qual é a **quantidade de itens** de um array

```
const pokemon = ["bulbasaur", "squirtle", "charmander"]
console.log(pokemon.length) // 3
```



EX2:

Cada personagem de desenho animado preparou seu lanche favorito. Temos as listas de lanches de quatro personagens: Homer Simpson, Scooby-Doo, Magali e Bob Esponja. Os lanches devem ser representados em arrays:

lancheHomer: rosquinha, rosquinha, rosquinha

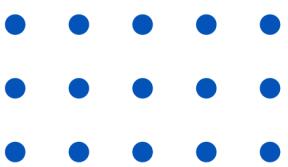
lancheScooby: hambúrguer, batata frita, milkshake, biscoito Scooby, pizza

lancheMagali: melancia, maçã, banana, abacate

lancheBob: hambúrguer de siri, refri, batata, sorvete, molho secreto

Utilize a propriedade **.length** com template strings para imprimir no console quantos itens tem o lanche de cada personagem.





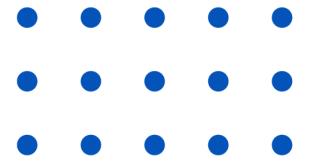
Método `includes(elemento)`



- O método `includes(elemento)` determina se um array contém um determinado elemento, retornando **true** ou **false**

```
const seriesBoas = ["Breaking Bad", "Brooklyn Nine-nine"]

seriesBoas.includes("Breaking Bad") // true
seriesBoas.includes("Game of Thrones") // false
```



EX3.

Você é um treinador Pokémon e tem uma lista dos Pokémons que já capturou:

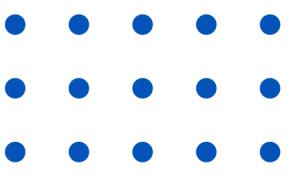
pokemonsCapturados:Pikachu, Charmander, Bulbasaur, Squirtle

Use o método **.includes()** para verificar se os seguintes Pokémons estão na sua lista:

"Pikachu"

"Meowth"





🍰 Desafio da Aula: Montando um Bolo em Camadas com Arrays

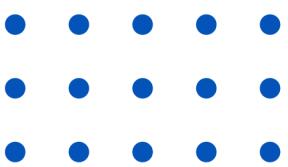
Nesta adorável unidade curricular, começamos a aprender o conceito de algoritmo como um passo a passo, como se fosse uma receita para fazer um bolo. E olha só... você está no caminho para se tornar um grande desenvolvedor, se desenvolvendo cada vez mais! 🎉

Agora, é hora de aplicar esse conhecimento de arrays em algo doce! 🍰

Instruções:

- Crie um array chamado **listaIngredientes** com todos os ingredientes necessários para fazer um bolo.
- Crie um array **baseDoBolo** com os ingredientes básicos para a massa do bolo.
- Crie três arrays diferentes para os recheios: **recheio1**, **recheio2**, **recheio3**.
- Crie um array chamado **boloPronto** que representa o bolo montado, alternando massa e recheio, como em camadas.
- Por fim, imprima (mostre) o conteúdo do bolo montado (acesse pelo índice da array).





Método push(elemento)

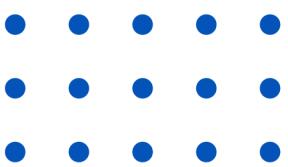
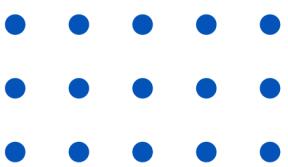


- O método **push(elemento)** adiciona um ou mais elementos ao final de um array

```
const numeros = [1, 2, 3]

numeros.push(4)
console.log(numeros) // [1, 2, 3, 4]

numeros.push(5, 6, 7)
console.log(numeros) // [1, 2, 3, 4, 5, 6, 7]
```



EX4

Você saiu para o mercado com uma lista inicial que sua mãe passou para você:

Lista De Compras: arroz, feijão, macarrão

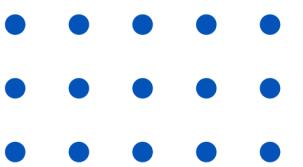
No meio do caminho, sua mãe te liga e pede para você adicionar mais itens à lista:

leite, farinha, banana, canela, café, frango, margarina e farofa para fazer o frango enfarofado da Erica

Use o método **.push()** para adicionar esses novos itens à lista.

Imprima a lista final no console.

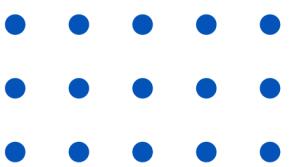




EX5

- Espada do Destino
- Poção de Vida
- Escudo da Coragem
- Elixir da Sabedoria
- Mapa do Desconhecido
- Amuleto do Guardião
- Botas da Agilidade
- Anel da Amizade
- Chave da Verdade
- Capa da Invisibilidade
- Lâmpada das Visões
- Flauta dos Espíritos
- Pergaminho das Runas
- Bracelete do Sacrifício
- Pedra do Renascimento

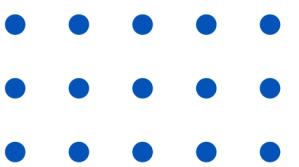




unshift() – Adicionar no começo

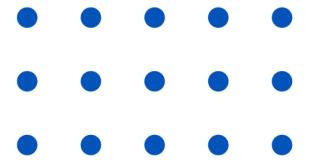
Adiciona um novo item no início do array.

```
let cores = ["azul", "verde"];
cores.unshift("vermelho");
console.log(cores); // ["vermelho", "azul", "verde"]
```



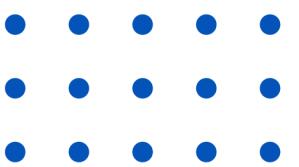
Ex7

Você tem uma lista com marcas de celular. Adicione "Motorola" no início da lista. Imprima a lista antes e depois de adicionar essa esplêndida marca.



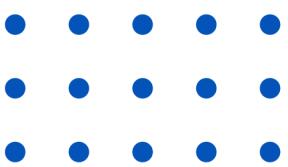
shift() – Remove do começo
Remove o primeiro item do array.

```
let frutas = ["maçã", "banana", "laranja"];
frutas.shift()
console.log(frutas); // ["banana", "laranja"]
```



Ex6:

Você tem uma lista com três cidades. Remova a primeira cidade da lista.



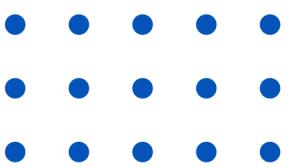
Método pop()



- O método `pop()` remove o último elemento de um array

```
const meusPeixes = ["palhaço", "mandarim", "esturjão"]
console.log(meusPeixes)

meusPeixes.pop()
console.log(meusPeixes) // ["palhaço", "mandarim"]
```

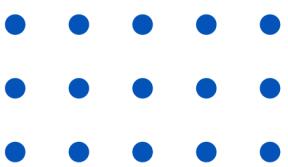


EX6.

Você está criando um gerenciador de tarefas simples em JavaScript. As tarefas são armazenadas em um array. Sempre que o usuário conclui uma tarefa, ele conclui a última adicionada (a mais recente), e ela deve ser removida usando o método `pop()`.

Implemente um código que:

1. Crie uma lista com 5 tarefas.
2. Mostre todas as tarefas.
3. Remova a última tarefa usando `pop()`.
4. Mostre qual tarefa foi concluída.
5. Mostre a lista atualizada.



Método splice(i, n)



- O método **splice(i, n)** remove **n** elementos à partir da posição **i** do array

```
const letras = ["A", "B", "C", "D", "E", "F", "G", "H"]
// índices (i)  0   1   2   3   4   5   6   7
console.log(letras)

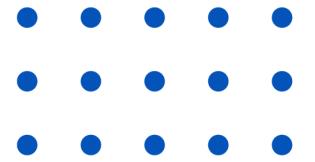
letras.splice(2, 1)
//      letras = ["A", "B", "D", "E", "F", "G", "H"]
// índices (i)      0   1   2   3   4   5   6
console.log(letras)

letras.splice(3, 2) // letras = ["A", "B", "D", "G", "H"]
console.log(letras)
```

-

da array `letras`, na posição 2 ("C") remove 1
(o próprio c)

da array `letras`, na posição 3 ("D") remove
2 (E e F)

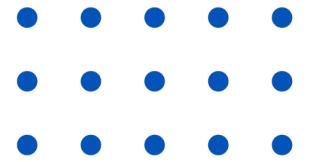


Você tem um deck de cartas Pokémon, mas percebe que algumas cartas estão na posição errada ou são repetidas.

```
let deck = ["Pikachu", "Charmander", "Bulbasaur", "Pikachu", "Squirtle", "Meowth"];
```

1. Remova a segunda carta "Pikachu" do deck (posição 3).
2. Insira a carta "Snorlax" entre "Bulbasaur" e "Squirtle".
3. Substitua a carta "Meowth" por "Eevee".

Use o método `.splice()` para todas as alterações.

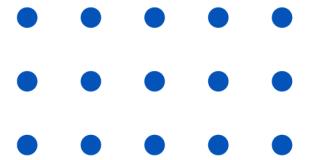


Splice para adicionar algo na Array

```
let frutas = ['maçã', 'banana', 'laranja'];

// Adicionando "kiwi" e "uva" na posição 1 (sem remover nada)
frutas.splice(1, 0, 'kiwi', 'uva');

console.log(frutas);
// Resultado: ['maçã', 'kiwi', 'uva', 'banana', 'laranja']
```



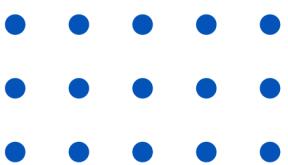
concat()

Concatenar significa juntar dois ou mais arrays para formar um novo array.

💡 Em JavaScript, arrays são estruturas de dados usadas para armazenar listas de elementos (como números, strings, objetos etc.).

```
let frutas2 = ["maçã", "banana"];
let legumes = ["cenoura", "batata"];

let alimentos = frutas2.concat(legumes);
console.log(alimentos); // ["maçã", "banana", "cenoura",
"batata"]
```

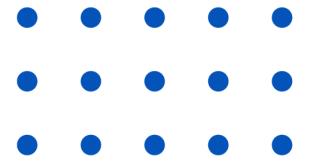


Ex.

Você abriu 3 pacotes de cartas Pokémon, e cada um veio com um conjunto diferente de cartas:

```
let pacote1 = ["Pikachu", "Bulbasaur", "Charmander"];
let pacote2 = ["Squirtle", "Jigglypuff"];
let pacote3 = ["Meowth", "Snorlax", "Eevee"];
```

- Crie uma nova lista chamada `cartasTotais` contendo todas as cartas dos 3 pacotes usando o método `.concat()`.
- Exiba no console a lista completa de cartas.
- Conte quantas cartas há no total e imprima essa informação.

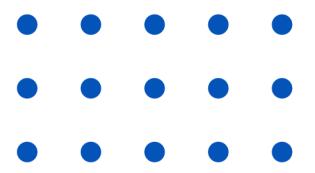


.sort() é um método do JavaScript que serve para ordenar os elementos de um array.

Como funciona?

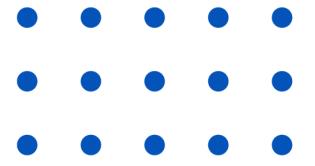
- Quando você chama `.sort()` em um array, ele organiza os itens em ordem crescente (por padrão, ordem alfabética para strings e ordem crescente para números, mas com algumas peculiaridades para números).

```
let frutas = ["Banana", "Maçã", "Laranja"];
frutas.sort();
console.log(frutas);
// Saída: ["Banana", "Laranja", "Maçã"]
```



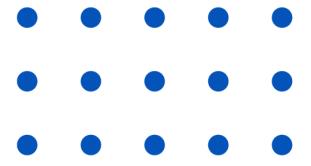
Trabalhando com vários métodos





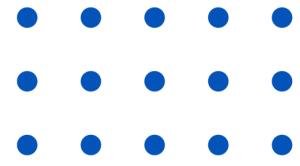
Você recebeu a seguinte frase: " Eu adoro JavaScript e estudar com meus colegas incríveis ! ". Faça o seguinte:

- Remova espaços no início e no fim da frase.
- Converta a frase para caixa baixa.
- Verifique se a frase inclui a palavra "javascript".
- Substitua todas as ocorrências da palavra "javascript" por "TypeScript".



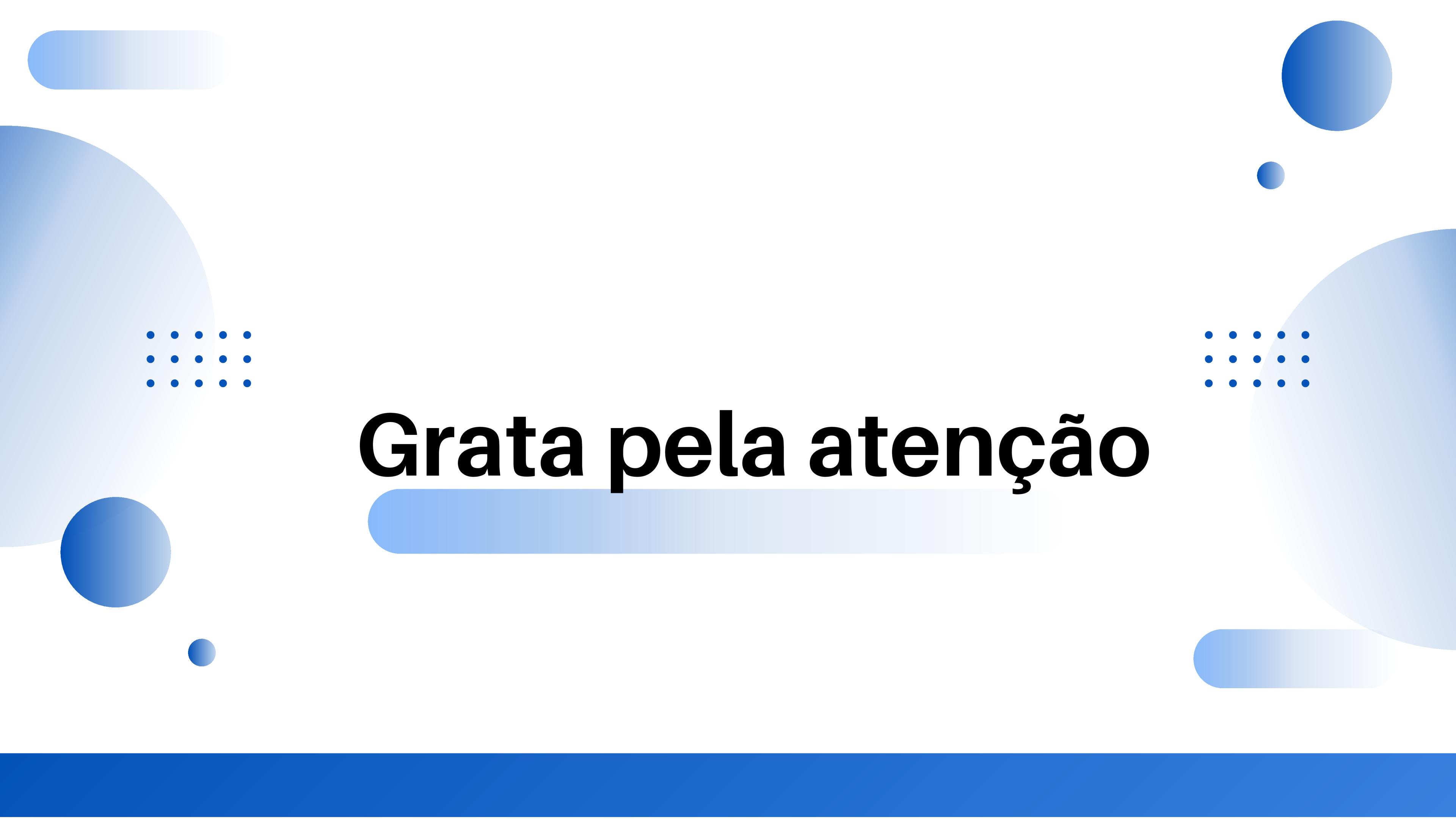
Dado o array de frutas = ["maçã", "banana", "laranja"], faça o seguinte:

- Adicione "morango" no final.
- Adicione "abacaxi" no começo.
- Remova o primeiro elemento.
- Remova o último elemento.
- Remova o elemento que está na posição 1 e adicione "manga" no lugar.



Dado um array de nomes = ["Ana", "Bruno", "Carlos"], faça:

- Converta todos os nomes para maiúsculas.
- Verifique se algum nome contém a letra "a".
- Adicione um novo nome "Lucas" ao final.
- Remova o nome do começo.
- Mostre a lista atualizada de nomes.



Grata pela atenção