

Loops

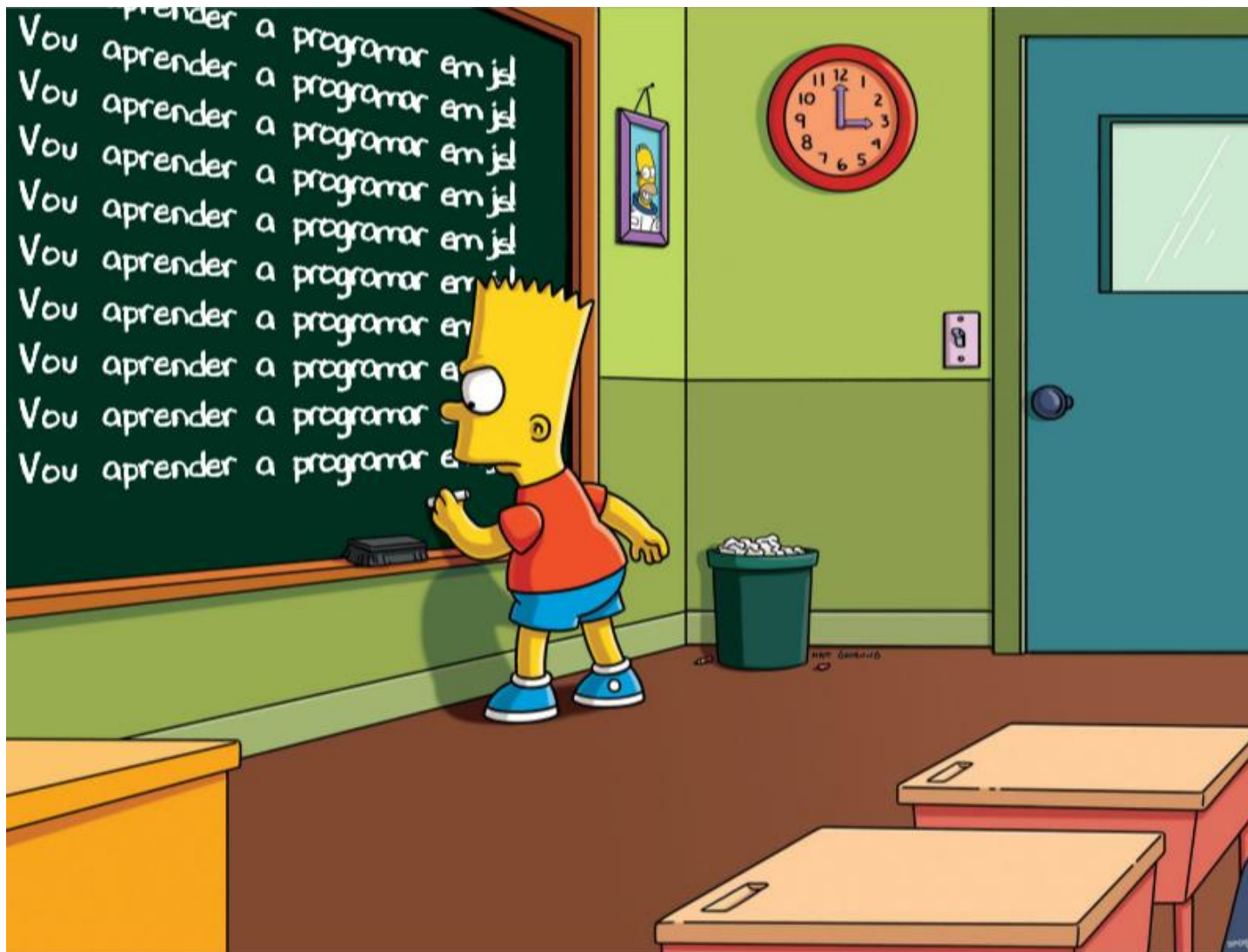
Igor N Faustino



O que são loops

- Loops são estruturas de controle que permitem que um programa execute uma ação repetidamente enquanto uma condição específica for verdadeira.
- Eles são usados para automatizar tarefas repetitivas em programação.
- Muito utilizados para manipular um conjunto de dados





Exemplo

- Um loop pode ajudar a gente a realizar esse tipo de tarefa

Repita 10 vezes:

Escreva("Vou aprender Javascript")



for

- Estrutura de repetição utilizada comumente quando sabemos quantas vezes precisaremos repetir uma ação
- Ex:
 - Repita X para cada número no conjunto
 - Repita X até chegar em 10
 - Repita X para cada letra em uma palavra



for



```
for (inicialização; condição; incremento) {  
    // código a ser executado  
}
```

- **Inicialização:** Executa uma vez antes da primeira iteração
- **Condição:** Avalia antes de cada iteração, Caso verdadeira, o Loop continua
- **Incremento:** Executa após cada iteração



Exemplo

- Contando números de 1 a 5



```
for (let i = 1; i ≤ 5; i++) {  
  console.log('Iteração número: ' + i);  
}
```



Exercício

- Calcule a soma dos números de 1 a 10 usando um loop `for`.



Exercício

- calcule e imprima a tabuada de multiplicação de um número



Exercício

- Escrever um algoritmo que gera e escreve os números ímpares entre 100 e 200.




Exercício

- Desenvolver um algoritmo que efetue a soma de todos os números ímpares que são múltiplos de três e que se encontram no conjunto dos números de 1 até 500.



While

- Loop é executado enquanto uma condição for verdadeira
- Útil para quando não conhecemos o número de iterações



```
while (condição) {  
    // código a ser executado enquanto a condição for  
    verdadeira  
}
```



Exemplo

- Executando 5 iterações com um loop while



```
let contador = 1;
while (contador ≤ 5) {
  console.log('Contador: ' + contador);
  contador++;
}
```



Exercício

- Dobre um número enquanto ele for menor do que 1000



do - while

- Similar ao loop while
- Mas garante a execução do loop **pelo menos 1 vez**
- **A condição só é verificada ao final da primeira iteração**



```
do {  
    // código a ser executado  
} while (condição);
```



Exemplo



```
let numero;  
do {  
    numero = prompt('Digite um número positivo:');  
} while (numero ≤ 0);  
console.log('Número válido: ' + numero);
```



Exercício

- Escreva um loop que simule o lançamento de um dado até que um número específico seja obtido e conte quantas tentativas foram necessárias.
- Para obter um número aleatório de 1 a 6:
 - **let resultado = Math.floor(Math.random() * 6) + 1;**



Loops Infinitos

- Um loop onde a condição de parada nunca é alcançada
- O loop vai continuar a ser executado indefinidamente
 - Pode estourar a memória do computador



Causas comuns

- **Erro na Condição:** A condição do loop nunca se torna falsa.
- **Incremento/Decremento ausente:** Esquecer de atualizar a variável de controle dentro do loop.
- **Condições sempre verdadeiras:** Definir uma condição que é sempre verdadeira, como `while(true)` sem um `break`.



Exemplo



```
let i = 0;  
while (i ≥ 0) {  
  console.log(i);  
  i++; // Isso nunca vai parar porque 'i' nunca será negativo  
}
```



Como evitar

- Certifique-se de que a condição do loop será eventualmente falsa.
- Verifique se o loop contém um mecanismo de saída, como um break ou um incremento/decremento adequado.



Uso Intencional

- Loops infinitos podem ser usados intencionalmente em programas que devem rodar continuamente, como servidores, mas devem sempre incluir uma forma segura de interrupção.



Exercício

- Escreva um algoritmo que receba valor inicial A e imprima seu fatorial.
 - Ex: $5! = 5 * 4 * 3 * 2 * 1 = 120$



Exercício

- Exiba no console o seguinte padrão utilizando loops

*

* *

* * *

* * * *



Exercício

- Exiba no console o seguinte padrão utilizando loops

```
* * *
```

```
* *
```

```
*
```



Exercício

- Exiba no console o seguinte padrão utilizando loops

*

* *

* * *

