



## Aula 2: Laços de Repetição

Estruturas fundamentais que permitem executar blocos de código repetidamente, otimizando nossa programação e tornando o código mais eficiente.

# O que veremos nesta aula



## Laços de repetição com for

Ideais quando sabemos exatamente quantas vezes queremos repetir uma ação



## Laços de repetição com while

Perfeitos para repetições baseadas em condições específicas

Estas estruturas são essenciais para automatizar tarefas repetitivas e criar programas mais dinâmicos e eficientes.

# Laço for e suas variações

O laço **for** é usado quando sabemos **quantas vezes queremos repetir** uma ação. Ele itera sobre uma sequência como range, lista ou string.

## Apenas fim

```
for i in range(5):  
    print(i)
```

**Saída:** 0, 1, 2, 3, 4

## Início e fim

```
for i in range(3, 7):  
    print(i)
```

**Saída:** 3, 4, 5, 6

## Início, fim e passo

```
for i in range(2, 11, 2):  
    print(i)
```

**Saída:** 2, 4, 6, 8, 10

### ? Atividade 1:

Crie um programa que conte de 0 até 10 com laço for.

### ? Atividade 2:

Crie um programa que conte de 10 até 1 com laço for, e ao final mostre "Feliz ano novo".



# Laço for em strings e listas

## Percorrendo strings

```
palavra = "Olá"  
for letra in palavra:  
    print(letra)
```

### Saída:

- O
- l
- á

## Percorrendo listas

```
frutas = ["maçã", "banana",  
          "uva"]  
for fruta in frutas:  
    print(fruta)
```

### Saída:

- maçã
- banana
- uva

Esta funcionalidade é extremamente útil para percorrer e processar todos os elementos em diferentes tipos de sequências.



### Atividade 3:

Crie um programa que peça ao usuário uma palavra e em seguida, solete esta palavra.



# Laço while (introdução)

O laço **while** é usado quando **não sabemos exatamente o número de repetições** necessárias. Ele executa enquanto a **condição for verdadeira**.

---

## Define a condição

A condição é avaliada antes de cada execução

---

## Executa o código

Se verdadeira, executa o bloco de código

---

## Reavalia a condição

Retorna ao passo 1 para nova verificação

```
contador = 0
while contador < 5:
    print(contador)
    contador += 1
```



# While infinito + break

⚠ **Cuidado:** Loops infinitos podem travar seu programa se não houver uma condição de saída!

```
contador = 0
while True:
    print(contador)
    contador += 1
    if contador == 5:
        break
```

**Saída:** 0, 1, 2, 3, 4

O comando **break** interrompe completamente a execução do laço.

## ❓ Atividade 4:

Crie um programa que conte de 0 até 10 com laço while.

## ❓ Atividade 5:

Crie um programa que conte de 10 até 1 com laço while, e ao final mostre "Feliz ano novo".



# Comando break

O **break** é usado para **parar a execução** do laço antes do seu final natural, sendo útil quando uma condição específica é atendida.



## Inicia o laço

```
for i in range(10):  
    if i == 5:  
        break  
    print(i)
```



## Break executa

Quando i chega a 5, o laço é interrompido imediatamente



## Resultado

Imprime apenas: 0, 1, 2, 3, 4

### ? Atividade 6:

Crie um programa que permita ao usuário digitar números, o programa deve ser encerrado apenas quando a somatória desses números ultrapassar 100. Ao final, mostre a soma total dos números.





# Comando continue

O **continue** pula apenas a iteração atual, continuando com a próxima iteração do laço.

## For + Continue

```
for i in range(6):  
    if i == 3:  
        continue  
    print(i)
```

**Resultado:** Pula o número 3

**Saída:** 0, 1, 2, 4, 5

## While + Continue

```
contador = 0  
while contador < 6:  
    contador += 1  
    if contador == 3:  
        continue  
    print(contador)
```

**Resultado:** Pula o número 3

**Saída:** 1, 2, 4, 5, 6

**Dica:** Use continue quando quiser pular certas condições, mas manter o laço executando.