

Aula 03

Estrutura de repetição - For



Tópicos

- Estruturas de repetição
- Comando de repetição “for”
- Estruturas de repetição aninhadas
- Comandos break e continue

Estrutura de Repetição

As estruturas de repetição

- Estruturas de repetição são usadas para executar um bloco de código várias vezes. Em Python, existem dois tipos principais de laços de repetição:
 - for e
 - while.

```
[5] for i in range(5):  
    print("Contador é:", i)
```

```
⇒ Contador é: 0  
   Contador é: 1  
   Contador é: 2  
   Contador é: 3  
   Contador é: 4
```

```
[2] # Exemplo prático  
    contador = 0  
  
    while contador < 5:  
        print("Contador é:", contador)  
        contador += 1
```

```
⇒ Contador é: 0  
   Contador é: 1  
   Contador é: 2  
   Contador é: 3  
   Contador é: 4
```

Estrutura de repetição “for”

Comando de repetição “for”

- O laço for é usado para iterar sobre uma sequência (como uma lista, tupla, string, ou range). É muito útil quando você sabe o número de iterações com antecedência.
- Uma instrução “break” executada no primeiro conjunto termina o loop sem executar o conjunto da cláusula else.
- Uma instrução “continue” executada no primeiro conjunto pula o resto do conjunto e continua com o próximo item, ou com a cláusula else se não houver próximo item.


Estruturas de repetição aninhadas

O “for” aninhado

- O laço “for” pode ser aninhado dentro de outros laço “for” ou “while”, permitindo a criação de estruturas de repetição mais complexas.
- O exemplo ao lado demonstra um laço for aninhado que utiliza duas variáveis (num e letra) para iterar sobre dois conjuntos de números e letras, formando pares de todos os elementos de ambos os conjuntos.

```
[14] # Definindo dois conjuntos de números
conjunto1 = [1, 2, 3]
conjunto2 = ['a', 'b', 'c']

# Trabalhando a estrutura for de maneira aninhada
for num in conjunto1:
    for letra in conjunto2:
        print(num, letra)
```



```
1 a
1 b
1 c
2 a
2 b
2 c
3 a
3 b
3 c
```


O “while” e o “for” aninhados

- Este exemplo mostra como combinar um laço “for” dentro de um laço “while” para realizar uma operação específica. Aqui, para cada número na lista números, o laço “while” controla a iteração sobre a lista, enquanto o laço “for” interno realiza uma contagem decrescente a partir do número atual.

```
# Lista de números
numeros = [3, 2, 1]

# Laço while externo
indice = 0
while indice < len(numeros):
    num = numeros[indice]

    # Laço for interno
    for i in range(num, 0, -1):
        print(f"Contando a partir de {num}: {i}")

    indice += 1
```

```
Contando a partir de 3: 3
Contando a partir de 3: 2
Contando a partir de 3: 1
Contando a partir de 2: 2
Contando a partir de 2: 1
Contando a partir de 1: 1
```


Comandos break e continue

Comandos break e continue


- Ambos os comandos são úteis para controlar o fluxo nos laços “while” e “for”, permitindo a implementação de lógica condicional para saída antecipada ou para pular partes do código dentro do laço.

Trabalhando com a estrutura “for”


```
[16] # Sai do laço quando i for 5
      for i in range(10):
          if i == 5:
              break
          print(i)
```



```
0
1
2
3
4
```



```
# Pula a iteração atual se i for par
for i in range(10):
    if i % 2 == 0:
        continue
    print(i)
```



```
1
3
5
7
9
```

Vamos praticar!
[Link](#)



Vamos exercitar!

[Link](#)

