Programação Aplicada ao Direito

Controle de fluxo de execução

Prof. Eduardo Mangeli

Decisões

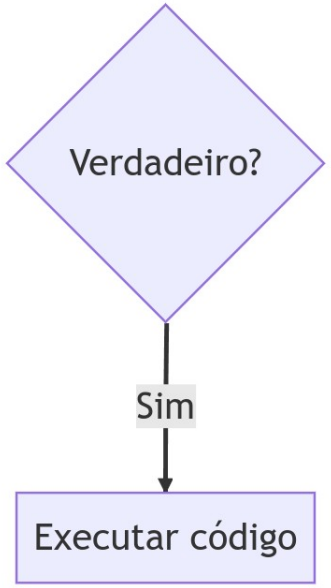
Mudando o fluxo de execução

A importância das estruturas de tomada de decisão reside na capacidade de controlar o fluxo de execução do programa.

Com base nas condições definidas, você pode orientar seu algoritmo para diferentes caminhos, executando diferentes instruções ou tomando diferentes decisões com base nos dados disponíveis.

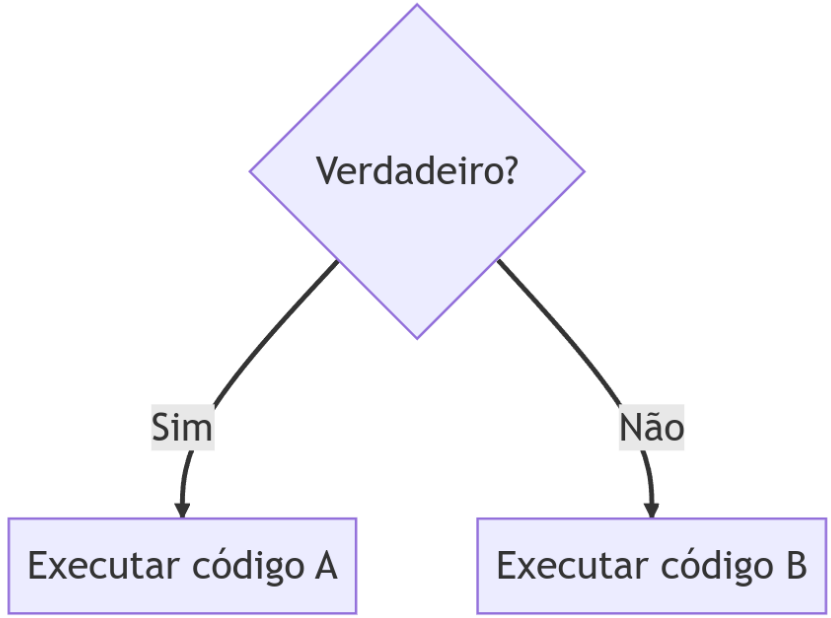
Estrutura if

```
idade = 18
if idade >= 18:
    print("Você é maior de idade!")
```



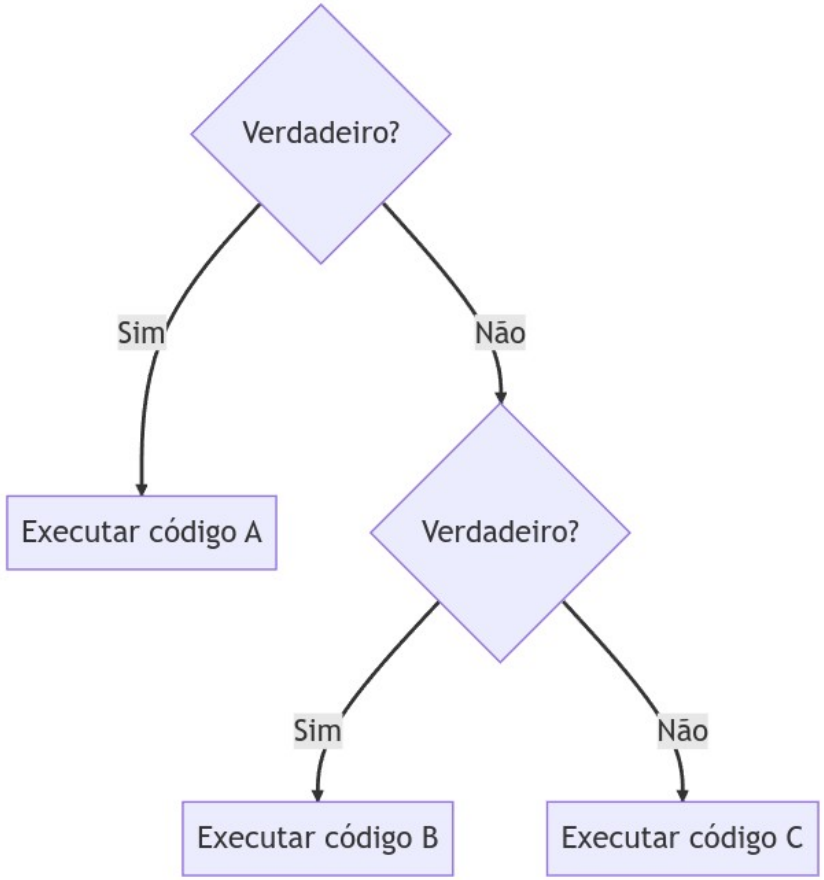
Estrutura if-else

```
idade = 16
if idade >= 18:
    print("Você é maior de idade!")
else:
    print("Você é menor de idade!")
```



Estrutura if-elif-else

```
idade = 25
if idade < 18:
    print("Você é menor de idade!")
elif idade >= 18 and idade < 60:
    print("Você é adulto!")
else:
    print("Você é um idoso!")
```



Operadores Lógicos

A	**B**	A and B	A or B	not A
False	**False**	False	False	True
False	**True**	False	True	True
True	**False**	False	True	False
True	**True**	True	True	False

Exemplo operador AND

```
idade = 22
tem_carteira = True
if idade >= 18 and tem_carteira:
    print("Você pode dirigir!")
else:
    print("Você não pode dirigir!")
```

Exemplo operador OR

```
idade = 16
tem_autorizacao = True
if idade >= 18 or tem_autorizacao:
    print("Você pode entrar na festa!")
else:
    print("Você não pode entrar na festa!")
```

Exemplo operador NOT

```
idade = 14
if not idade >= 18:
    print("Você é menor de idade!")
```

Ordem de Precedência

- not, and, or
  - Minemônico: Não
- Podemos usar parênteses

Laços

O que são laços em Programação?

Os laços, também conhecidos como loops, são uma estrutura fundamental em programação que nos permitem repetir uma determinada sequência de instruções várias vezes. Eles são essenciais para automatizar tarefas e lidar com conjuntos de dados.

Em Python, existem dois tipos principais de laços:

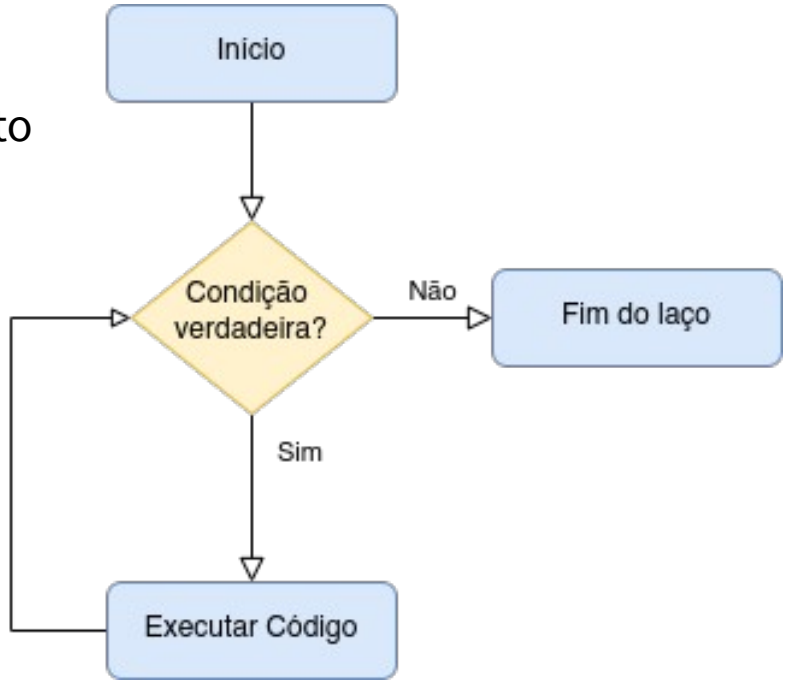
- o laço **while**, e

- o laço **for**.

Laço while

O laço while é usado quando não sabemos antecipadamente o número exato de vezes que uma determinada ação deve ser repetida. Ele continua executando um bloco de código enquanto uma condição especificada for verdadeira.

```
while condição:
    # código a ser executado
```

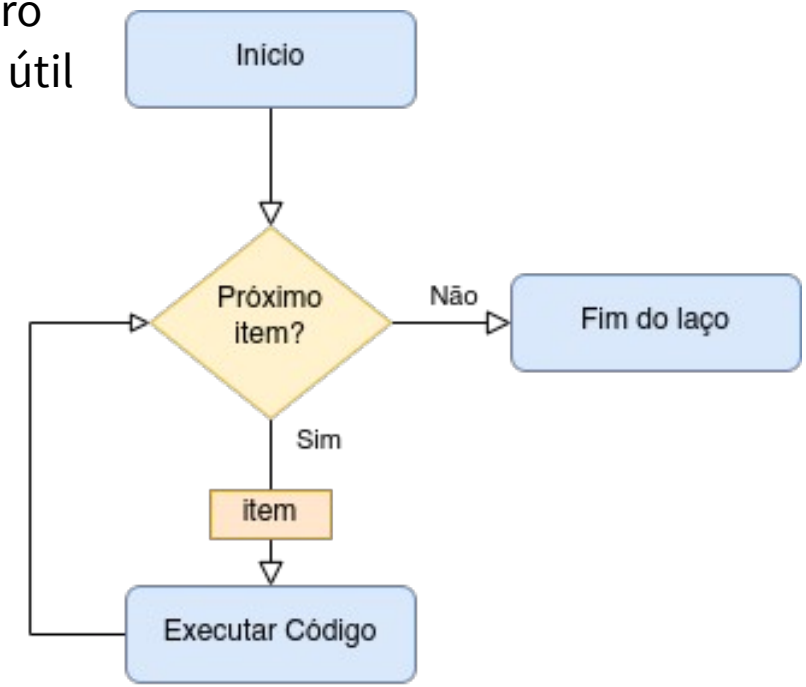


O fluxo de execução começa com a avaliação da condição. Se a condição for verdadeira, o bloco de código dentro do laço é executado. Depois, a condição é avaliada novamente. Esse processo se repete até que a condição se torne falsa.

Laço for

O laço for é usado quando sabemos antecipadamente o número de vezes que uma ação deve ser repetida. Ele é especialmente útil quando queremos iterar sobre uma sequência de elementos, como uma lista, uma string ou um intervalo de números.

```
for item in sequência:
    # código a ser executado
```



O fluxo de execução do laço for é controlado pela sequência. A cada iteração, um elemento da sequência é atribuído à variável **item**, e o bloco de código dentro do laço é executado. Esse processo se repete até que todos os elementos da sequência sejam percorridos.

Importância dos laços

Os laços desempenham um papel crucial na computação e são amplamente utilizados por várias razões:

1. **Automatização de tarefas repetitivas:** Os laços permitem executar um conjunto de instruções várias vezes, automatizando tarefas que, de outra forma, seriam tediosas e propensas a erros.
2. **Processamento de conjuntos de dados:** Com os laços, podemos percorrer e manipular elementos de uma lista, string ou qualquer outra estrutura de dados, facilitando a realização de operações em massa.
3. **Iteração em coleções:** Ao utilizar laços for, podemos iterar sobre cada elemento de uma sequência, como uma lista, sem a necessidade de acompanhar manualmente o índice atual.
4. **Tomada de decisões:** Os laços podem ser usados em combinação com instruções condicionais (como if) para tomar decisões com base em determinadas condições e realizar diferentes ações em diferentes situações.