Blocos de Comandos

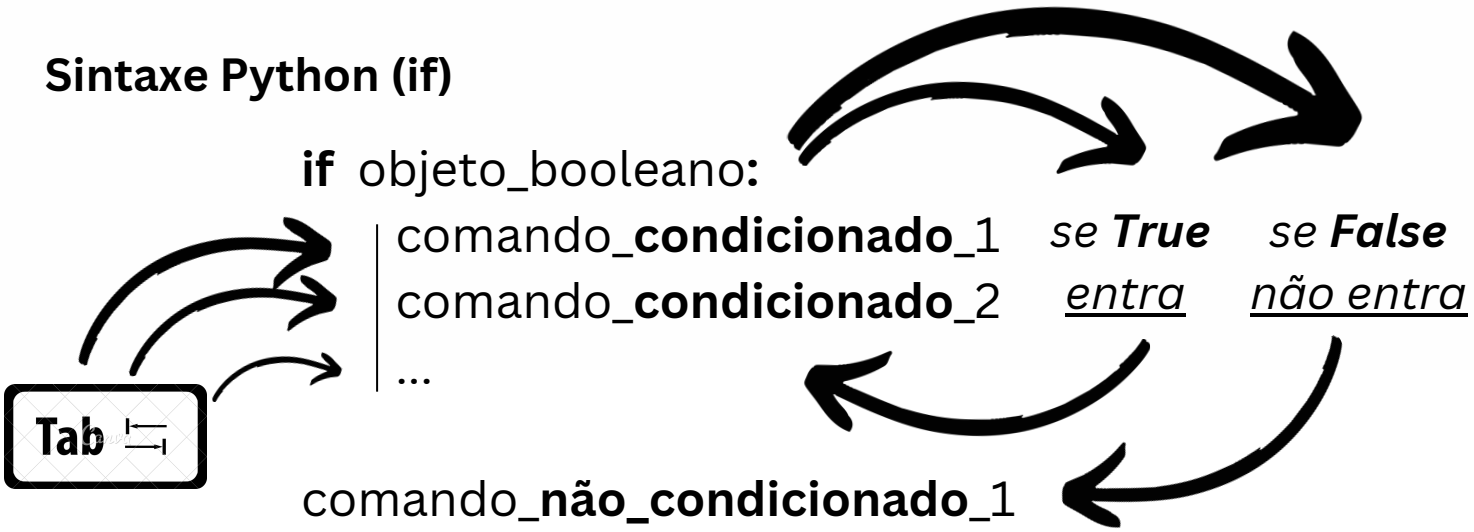
Conceito

Os blocos de comandos são **instruções** para fazer que comandos sejam contidos dentro de **contextos**. Esses blocos permitem quebrar a sequência linear (de cima para baixo, uma linha por vez) de execução. Para que comandos sejam atrelados a contextos, é necessário que estes comandos estejam abaixo de uma instrução **if**, **else**, **elif**, **while**, **for**, etc., e conter um espaçamento (indentação) a mais que as instruções mencionadas acima. Os espaçamentos (indentações) adicionados devem ser observados cuidadosamente para adequar corretamente o contexto de cada comando. Esses espaçamentos são aplicados pela tecla Tab.

Principais instruções de blocos de comandos

- if Bloco condicional
- elif Bloco condicional extra
- else Bloco condicional final
- for Bloco de repetição por coleção
- while Bloco de repetição condicional

Sintaxe Python (if)



Blocos de Comandos

Observações (if)

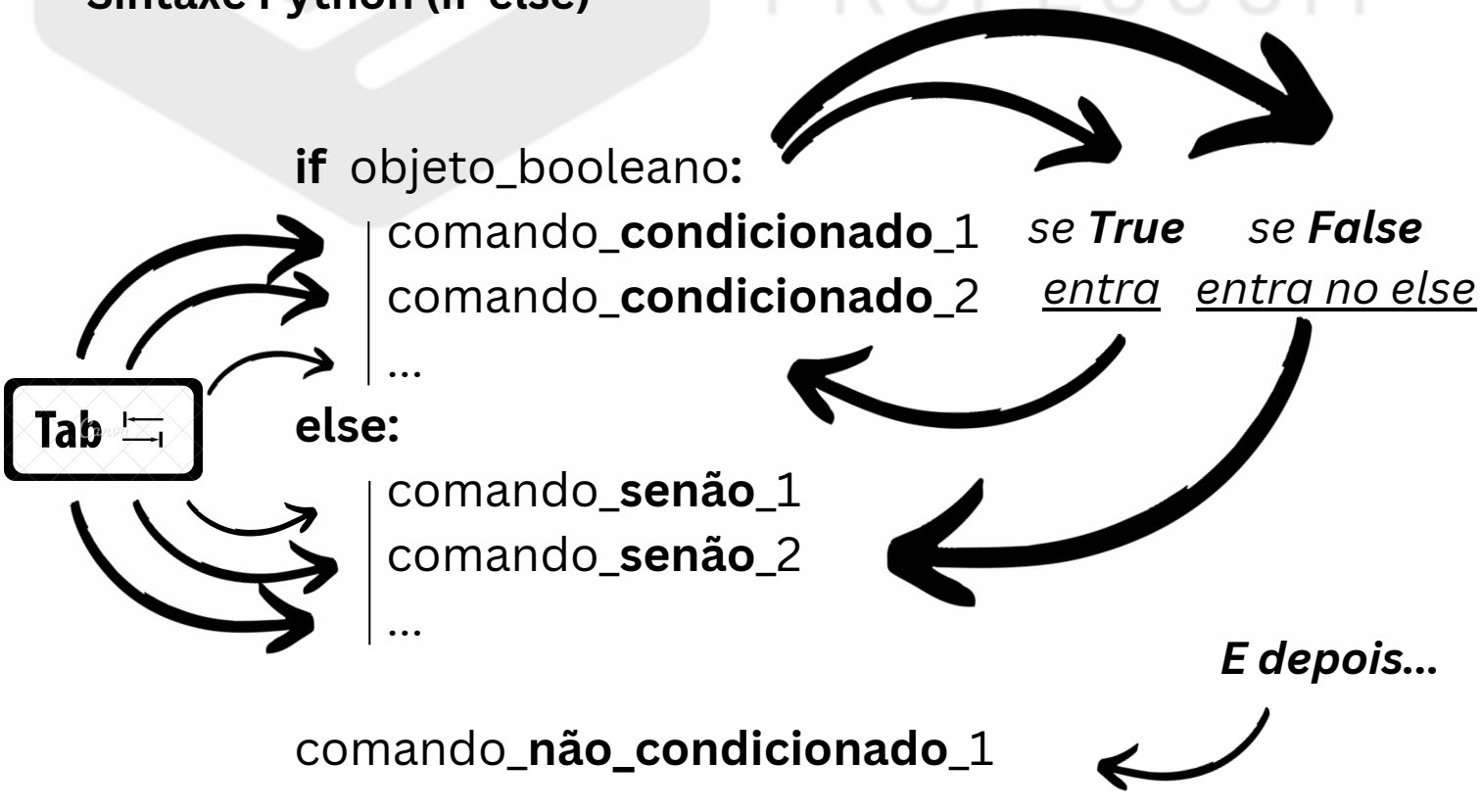
1. Perceber que o comando_**não_condicionado** possui a **mesma (fora)** indentação que a instrução if acima, enquando os comandos_**condicionados** estão com um **tab a mais (dentro)** que a instrução if acima.
2. O objeto booleano pode ser obtido através de uma **expressão lógica** ou simplesmente de uma variável que armazene um objeto booleano.

EXAMPLE condicao_1 **and** condicao_2 → objeto_booleano 1

EXAMPLE condicao_1 **or** condicao_2 → objeto_booleano 2

EXAMPLE **(**condicao_1 **or** condicao_2**) and** condicao_3 → objeto_booleano 3

Sintaxe Python (if-else)



Blocos de Comandos

Observações (if-else)

1. O bloco else é **estritamente** atrelado ao bloco if. Ou seja, é impossível haver else **sem** if. O inverso é possível.
2. Se o objeto_booleano for **True**, a execução **entra** no bloco if, se **não for True** (ou seja, False), a execução **'pula'** os comandos do if e parte **diretamente** para os comandos do **else**.
3. **Após** a execução dos comandos if **ou** else, a execução parte para o(s) comando(s) **não condicionados**.

Sintaxe Python (if-elif-else)

```
if objeto_booleano_1:
    comando_condicionado_1__1
    comando_condicionado_1__2
    ...
elif objeto_booleano_2:
    comando_condicionado_2_1
    comando_condicionado_2_2
    ...
elif objeto_booleano_3:
    comando_condicionado_2_1
    comando_condicionado_2_2
    ...
else:
    comando_senão_1
    comando_senao_2
    ...
comando_não_condicionado_1
...
```

Tab

*se **True** entra*

*se **False** **checa** o próximo elif*

*se **True** entra*

*se **False** **checa** o próximo elif*

*se **True** entra*

*se **False** entra else*

E depois...

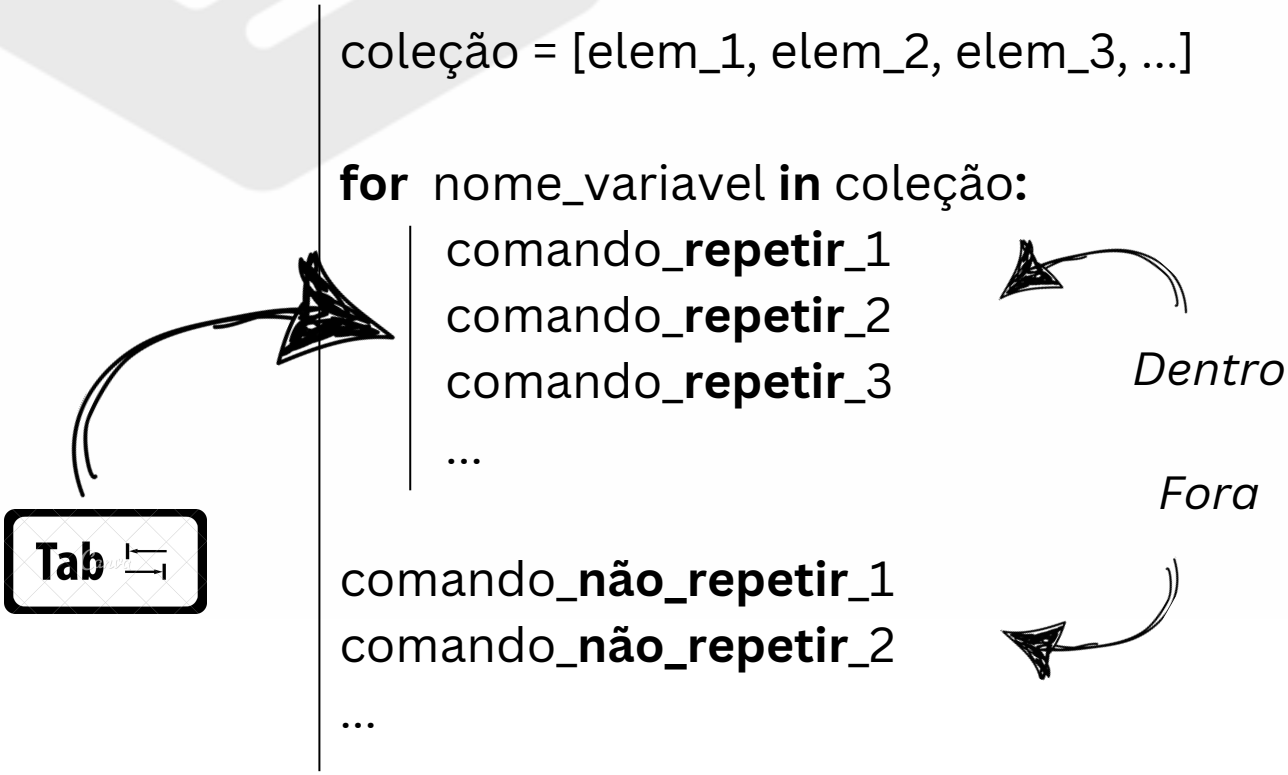
Curso de Python 1

Blocos de Comandos

Observações (if-elif-else)

1. O bloco elif é estritamente atrelado ao bloco if. Ou seja, é impossível haver elif sem if. O inverso é possível.
2. Se o objeto_booleano_1 for **True**, a execução **entra** no bloco if. Se for **False**, a execução **'testa'** o objeto booleano_2, e se for **True**, a execução **entra** no primeiro elif. Se for **False**, a execução **'testa'** o objeto_booleano_3, e se for True, a execução **entra** no segundo elif. Finalmente, se for **False**, a execução entra no bloco **else**.
3. Após a execução dos comandos if **ou** elif_1 **ou** elif2 **ou** else, a execução parte para o(s) comando(s) não condicionados.

Sintaxe Python (for)



Blocos de Comandos

Observações (for)

1. O bloco for **exige** a **criação** de uma **variável** (antes da instrução ‘in’), que por sua vez, vai **percorrer cada elemento** da coleção fornecida (após a instrução ‘in’).
2. Os comandos utilizados dentro do for, serão repetidos n vezes, sendo que, n é o tamanho (quantidade de elementos) da coleção. Essas repetições é conhecida, em programação, como **iterações**.
3. Cada iteração, a variável **criada** (nome_variavel no exemplo acima) será **utilizada** para servir um propósito (lógica) maior.

Sintaxe Python (for)

Observações (while)

1. While é o **mesmo** que o bloco **if**, porém, a condição (objeto_booleano) é checado após **cada iteração** para direcionar a execução para **dentro** do bloco while **ou** para **fora** deste.
2. Os blocos while e for executam a mesma operação, de **maneiras distintas**.