Fundamentos da Linguagem Python II

Rodrigo Richard Gomes

Controle de Fluxo

Comando condicional
if, if..else, if..elif

Comando condicional – if

```
if (condição):
    bloco_comandos1
```

Comando condicional – if

```
if (condição):
    bloco_comandos1
```

Comando condicional – if

```
if (condição):
    bloco_comandos1
```

A identação (ou recuo) deve ter 4 espaços ou 1 tabulação

Comando condicional – if

```
if (condição):
    bloco_comandos1
```

Condição **VERDADEIRA**
O $bloco\_comandos_1$ é executado

Comando condicional – if

Condição **FALSA**
O bloco_comandos$_1$ não é executado

```
if (condição):
    bloco_comandos1
```

Comando condicional – if..else

```
if (condição):
    bloco_comandos1
else:
    bloco_comandos2
```

Comando condicional – if..else

Condição **VERDADEIRA**
O bloco_comandos$_1$ é executado

```
if (condição):
    bloco_comandos1
else:
    bloco_comandos2
```

Comando condicional – if..else

Condição **FALSA**
O bloco_comandos$_2$ é executado

```
if (condição):
    bloco_comandos1
else:
    bloco_comandos2
```

Comando condicional – if..elif

```
if (condição_1):
    bloco_comandos_1
elif (condição_2):
    bloco_comandos_2

...
elif (condição_n):
    bloco_comandos_n
```

Comando condicional – if..elif

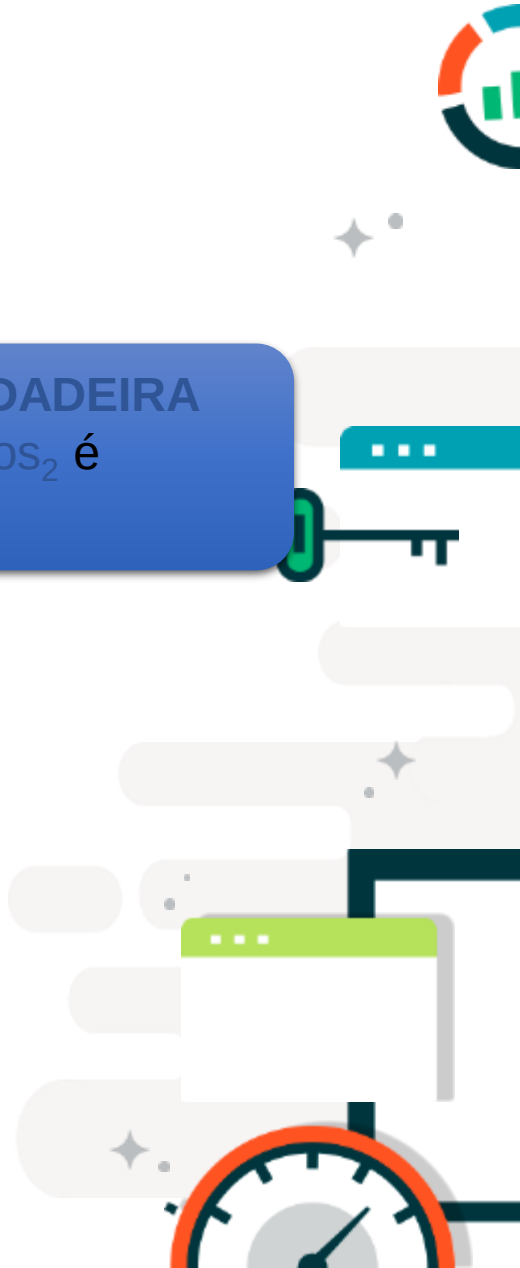
```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
```

Condição 1 **VERDADEIRA**
O $bloco\_comandos_1$ é executado

Comando condicional – if..elif

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
```

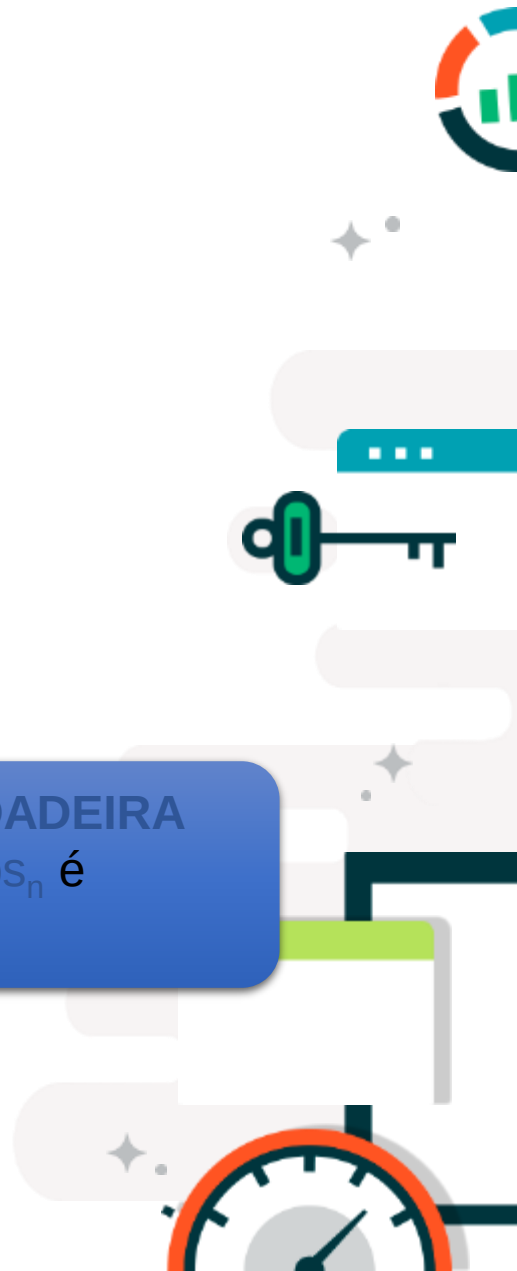
Condição 2 **VERDADEIRA**
O $bloco\_comandos_2$ é executado



Comando condicional – if..elif..else

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
```

Condição n **VERDADEIRA**
O bloco_comandos$_n$ é executado



Comando condicional – if..elif..else

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2
...
elif (condiçãon):
    bloco_comandosn
else:
    bloco_comandoselse
```

Comando condicional – if..else..elif

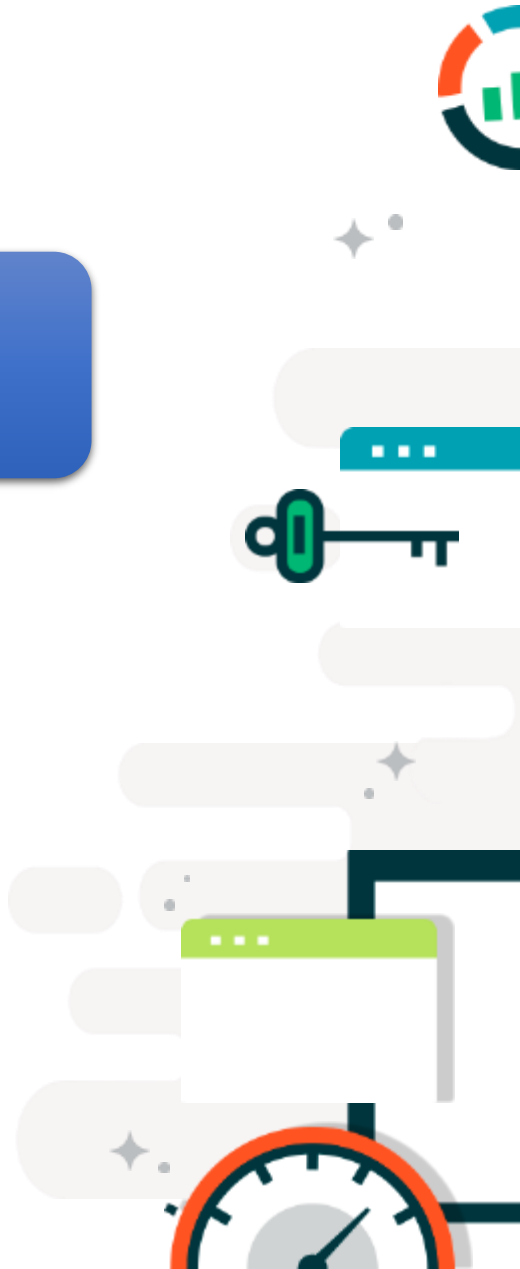
```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
else:
    bloco_comandoselse
```

Condição 1 **VERDADEIRA**
O $bloco\_comandos_1$ é executado

Comando condicional – if..elif..else

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2
...
elif (condiçãon):
    bloco_comandosn
else:
    bloco_comandoselse
```

Condição 2 **VERDADEIRA**
O $bloco\_comandos_2$ é executado



Comando condicional – if..elif..else

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
else:
    bloco_comandoselse
```

Condição n **VERDADEIRA**
O $bloco\_comandos_n$ é executado

Comando condicional – if..elif..else

```
if (condição1):
    bloco_comandos1
elif (condição2):
    bloco_comandos2

...
elif (condiçãon):
    bloco_comandosn
else:
    bloco_comandoselse
```

Nenhuma das condições anteriores **VERDADEIRA**
O $bloco\_comandos_{else}$ é executado

Comando condicional – if

- Uma condição em um comando **if** é qualquer expressão lógica que retorna **True** ou **False**
- Pode-se usar os seguintes operadores relacionais

Operador	Significado	Exemplo
==	Igualdade	a == b
!=	Desilgualdade	a != b
>	Maior que	a > b
<	Menor que	a < b
>=	Maior ou igual	a >= b
<=	Menor ou igual	a <= b

Comando condicional – if

- Pode-se utilizar condições compostas unidas por conectivos lógicos

Operador	Significado	Exemplo
and	**E** lógico	a == b and b == c
or	**OU** lógico	a == b or b == c
^	**OU** exclusivo (XOR)	a == b ^ b == c
not	Negação (inverte o resultado lógico)	not(a == b and b == c)

Operadores lógicos (conectivos)

Exemplos	

`if (num < 0):`
`    print("O valor da variável num é negativo")`
	`if (num < 0):`
	`    print("O valor da variável num é negativo")`
	`else:`
	`    print("O valor da variável num é positivo")`

`if (num < 0):`
`    print("O valor da variável num é negativo")`
`else:`
`    if (num == 0):`
`        print("O valor da variável num é nulo")`
`    else:`
`        print("O valor da variável num é positivo")`
	`if (num < 0):`
	`    print("O valor da variável num é negativo")`
	`elif (num == 0):`
	`    print("O valor da variável num é nulo")`
	`else:`
	`    print("O valor da variável num é positivo")`

`if (num >= 0) and (num % 2 == 0):` `    print("O valor da variável num é positivo e par")`	

Comando de repetição while

Comando de repetição – while

```
while (condição):
    bloco_comandos
```

Comando de repetição – while

```
print("Para interromper a execução digite número <= 0")
num = int(input("Digite um número: "))
while num > 0:
    raiz = num ** (1/2)
    print("Raiz quadrada de {} = {}.".format(num, raiz))
    num = int(input("Digite um número: "))
print("Fim de execução.")
```

Comando de repetição – while

```
qtde = 1
while qtde <= 5:
    num = int(input("Digite um número: "))
    if num > 0:
        raiz = num ** (1/2)
        print(“Raiz quadrada de {} = {}.".format(num, raiz))
    qtde = qtde + 1
print("Fim de execução.")
```

Comando de repetição for

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val ☐

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val | 15 |

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val | 8 |

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val | 12 |

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val | 19 |

Comando de repetição – for

```
for val in sequencia:
    bloco_comandos
```

15	8	12	19	5
0	1	2	3	4

val | 5 |

Comando de repetição – for

```
lista = [15, 8, 12, 19, 5]
for num in lista:
    print(num)
```

Comando de repetição – for

Simulando o comando *for* de linguagens *C-like* usando a função *range*()

```
range([inicio], final [, passo])
```

Exemplos:

`range(5)` → 0, 1, 2, 3, 4

`range(2, 8)` → 2, 3, 4, 5, 6, 7

`range(5, 35, 7)` → 5, 12, 19, 26, 33

Comando de repetição – for

Simulando o comando *for* de linguagens *C-like* usando a função *range*()

Comando for em Java

```
for (int i = 0; i < 10; i++)
    ...
```

Equivalente em Python

```
for i in range(10):
    ...
```

Comando de repetição – for

Simulando o comando *for* de linguagens *C-like* usando a função *range*()

Comando for em Java

```
for (int i = 10; i < 15; i++)
    ...
```

Equivalente em Python

```
for i in range(10, 15):
    ...
```

Comando de repetição – for

Simulando o comando *for* de linguagens *C-like* usando a função *range*()

Comando for em Java

```
for (int i = 10; i < 50; i += 5)
    ...
```

Equivalente em Python

```
for i in range(10, 50, 5):
    ...
```

Funções

```
def nome_funcao(parametros):
    """ Documentation String """
    comando1
    comando2
    ...
    comandon
```

Funções

```
def nome_funcao(parametros):
    """ Documentation String """
    comando_1
    comando_2
    ...
    comando_n
    return # opcional
```