Operadores de comparação e lógicos

Dados lógicos (`True` , `False`) aparecem com mais frequência junto a **operadores relacionais** ou **de comparação** (ou seja, operadores que checam se uma equivalência entre dados -- sejam eles `int` , `float` , `str` etc. -- é *verdadeira* ou *falsa*.

Estes são os operadores de comparação em Python (e nos exemplos, `x = 5`):

operador	significado	`input` de exemplo	`output`
==	igual a	`x == 1`	`False`
!=	não-igual a	`x != 2`	`True`
>	maior que	`x > 3`	`True`
>=	maior que ou igual a	`x >= 6`	`False`
<	menor que	`x < 3`	`False`
<=	menor que ou igual a	`x <= 5`	`True`

In [1]:
```
print(5 ** 2 / 2)
```

```
12.5
```

In [2]:
```
# O resultado da equação é igual a 12?
print(5 ** 2 / 2 == 12)
```

```
False
```

In [3]:
```
# O resultado da equação é maior que ou igual a 12?
print(5 ** 2 / 2 >= 12)
```

```
True
```

In [4]:
```
frase = "Boa noite a todos!"
print(frase)
print(len(frase))  # len() retorna o número de itens num objeto; uma frase
print(len(frase) < 10)
```

```
Boa noite a todos!
18
False
```

In [5]:
```
# Lembra quando eu disse que False é 0 e True é 1?
print(True == 1)
print(False == 0)
print(True + True)
```

```
True
True
2
```

Além dos operadores de comparação, há os operadores lógicos em Python (e nos exemplos, `x = 5`):

operador	significado	input de exemplo	output
and	e lógico (ambas as comparações)	x > 1 and x < 10	True
or	ou lógico (uma das comparações)	x != 2 or x != 5	True
not	negação lógica (inverte a resposta lógica)	not x == 5	False

In [6]:
```
# Determinamos que a variável x representa o número 5
x = 5
```

In [7]:
```
# "x é maior que 1 **E** menor que 10?"
# (As duas comparações precisam ser True para o resultado ser True)
x > 1 and x < 10
```

Out[7]: True

In [8]:
```
# "x é diferente de 2 **OU** diferente de 5?"
# (Uma das duas comparações precisa ser True para o resultado ser True)
x != 2 or x != 5
```

Out[8]: True

In [9]:
```
# "Qual é o **OPOSTO** da afirmação "x é igual a 5"?"
# (se x é igual a 5, o resultado será False; se x não for igual a 5, o res
not x == 5
```

Out[9]: False

Controle de fluxo com if-elif-else

Até o momento, estamos escrevendo códigos que são executados numa estrutura fixa: de cima para baixo, linha a linha. Exemplo:

```
[linha 1] nome = input("Digite seu nome :")
[linha 2] ano = input("Digite o ano de nascimento: ")
[linha 3] idade = 2021 - int(y)
[linha 4] print("Bom dia, {}. Você tem ou terá {} anos em
2021.".format(nome, str(idade)))
```

Até aqui, nosso script processa a linha 1, depois a linha 2, depois a linha 3 e, por fim, a linha 4.

É um fluxo fixo. Mas podemos mudar isso. Podemos **controlar** a execução, o fluxo do script.

Uma das formas é com `if-elif-else` , que **condiciona** a execução de uma linha ao resultado de outra. Sua extrutura é assim:

```
if condicao_1:
    acao_1
elif condicao_2:
    acao_2
```

```
elif condicao_n:
    acao_n
else:
    acao_quando_nenhuma_das_condicoes_foi_preenchida
```

Por exemplo: *quero imprimir se a idade do usuário é par ou ímpar*. Algo mais ou menos assim:

```
[linha 1] se o resto da divisão da idade por 2 for 0,
[linha 2] imprima "é par"
[linha 3] caso contrário,
[linha 4] imprima "é ímpar"
```

Repare que, no exemplo acima, apenas **uma** das duas impressões (`é par` ou `é ímpar`) será executada, a depender do resultado do resto da divisão da idade (`0` ou `1`). **A outra impressão não será executada**.

E como podemos fazer isso? Bem, já sabemos como ver se o resto de uma divisão é `0` :

In [10]:
```
idade_1 = 24
print(idade_1 % 2 == 0)
```

```
True
```

In [11]:
```
idade_2 = 37
print(idade_2 % 2 == 0)
```

```
False
```

Também sabemos imprimir:

In [12]:
```
print("É par.")
```

```
É par.
```

In [13]:
```
print("É ímpar.")
```

```
É ímpar.
```

O que precisamos agora é **condicionar** a impressão de acordo com o resultado. Para traduzir o exemplo acima em Python, fica assim:

```
if idade_x % 2 == 0:
    print("É par.")
else:
    print("É ímpar.")
```

In [14]:
```
idade_x = 12

if idade_x % 2 == 0:
    print("É par.")
else:
    print("É ímpar.")
```

```
É par.
```

In [15]:

```
idade_x = 13

if idade_x % 2 == 0:
    print("É par.")
else:
    print("É ímpar.")
```

```
É ímpar.
```

Ok, vamos tentar algo mais complexo e sua estrutura:

In [16]:

```
temp = input("Qual é a temperatura agora? ")

if int(temp) <= 17:
    print("{} graus significa que está frio!".format(temp))
elif int(temp) > 17 and int(temp) <= 26:
    print("{} graus significa que está um tempo ameno.".format(temp))
elif int(temp) > 26 and int(temp) <= 38:
    print("{} graus significa que está um tempo quente.".format(temp))
else:
    print("{} graus significa inferno na terra!".format(temp))
```

```
Qual é a temperatura agora? 27
27 graus significa que está um tempo quente.
```