Seleção condicional

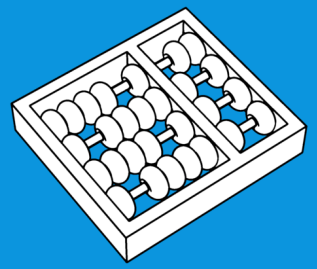
MC102 - Algoritmos e Programação de Computadores

03/25

4

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP



"O interesse da matemática não é ser familiarizada com ideias de números e quantidades."

George Boole.

Dúvidas da aula anterior

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?
- É possível usar os operadores comparativos com strings?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?
- É possível usar os operadores comparativos com strings?
- Em python, é possível obter um valor booleano como resultado da comparação entre dois dados também do tipo bool? (p. ex. se x = False e y = False, então a operação comparativa x == y resultaria num terceiro dado booleano igual a True?)

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?
- É possível usar os operadores comparativos com strings?
- Em python, é possível obter um valor booleano como resultado da comparação entre dois dados também do tipo bool? (p. ex. se x = False e y = False, então a operação comparativa x == y resultaria num terceiro dado booleano igual a True?)
- A imprecisão dos floats decimais pode resultar em problemas ao comparar sua igualdade? EX: A = 0.8 - 0.2, B = 0.9-0.3, A == B?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?
- É possível usar os operadores comparativos com strings?
- Em python, é possível obter um valor booleano como resultado da comparação entre dois dados também do tipo bool? (p. ex. se x = False e y = False, então a operação comparativa x == y resultaria num terceiro dado booleano igual a True?)
- A imprecisão dos floats decimais pode resultar em problemas ao comparar sua igualdade? EX: A = 0.8 - 0.2, B = 0.9-0.3, A == B?
- É possível criar grandes aplicações de jogos com python?

Dúvidas selecionadas

- É possível escrever o código em qualquer lugar (por exemplo, no bloco de notas) ou é necessário usar um editor de texto, como o vs code?
- Como devem ser nossos comentários nos códigos que vão ser avaliados? Eles devem explicar cada passo?
- Dado um problema com várias soluções possíveis, existe uma forma de descobrir qual delas é mais eficiente ou é só tendo muita experiência/testando todas? Claro que em alguns problemas fica claro que tem um jeito melhor de resolver, mas em outros nem tanto.
- Para que serve a álgebra booleana além da programação?
- Não entendi como separar as entradas e saídas de um código por meio de <entrada.txt> saida.txt. Como fazer isso? (CMD cria um arquivo na minha área de trabalho chamado saida.txt com os outputs do código? Ou eu devo criar e dizer o diretório deste arquivo?)
- Em python, por que o input sempre retorna uma string, mesmo se a entrada é um número?
- O que acontece se eu não seguir a identação no código python?
- Qual o limite de quantidade de letras dentro de uma variável string?
- É possível usar os operadores comparativos com strings?
- Em python, é possível obter um valor booleano como resultado da comparação entre dois dados também do tipo bool? (p. ex. se x = False e y = False, então a operação comparativa x == y resultaria num terceiro dado booleano igual a True?)
- A imprecisão dos floats decimais pode resultar em problemas ao comparar sua igualdade? EX: A = 0.8 - 0.2, B = 0.9-0.3, A == B?
- É possível criar grandes aplicações de jogos com python?
- Nao entendi o caso do if dentro do if:

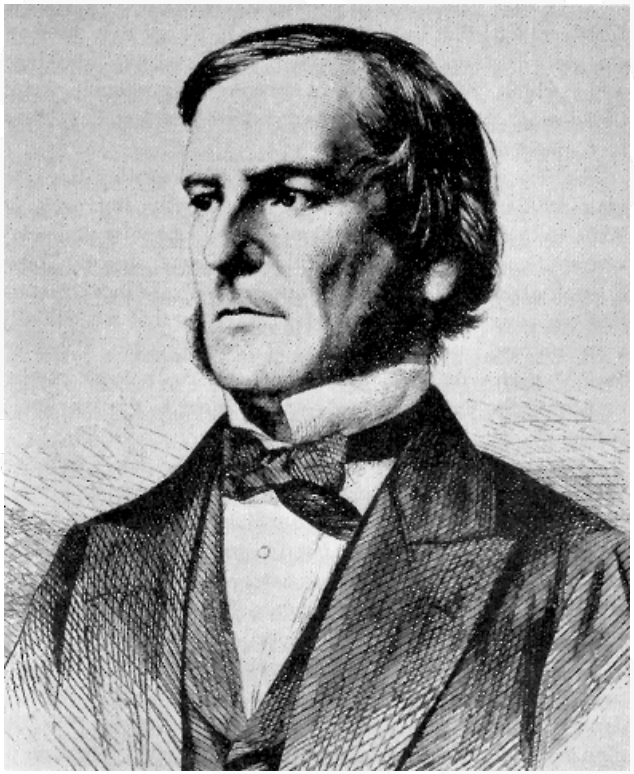
```
if:
   if:
   else:
else:
```

É a mesma coisa que esse embaixo ou é diferente?

```
if:
else:
if:
else:
```

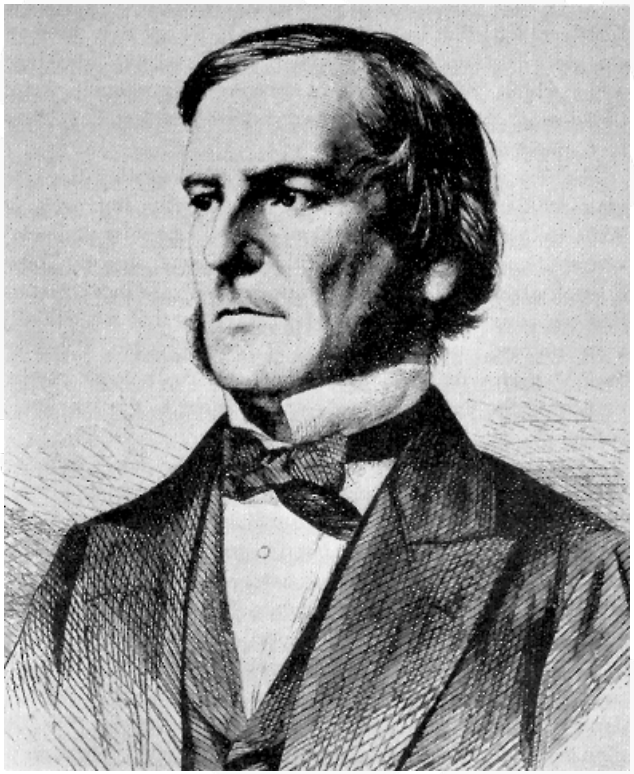
Álgebra booleana

George Boole (02/09/1815 – 08/12/1864)



Principais áreas de atuação:

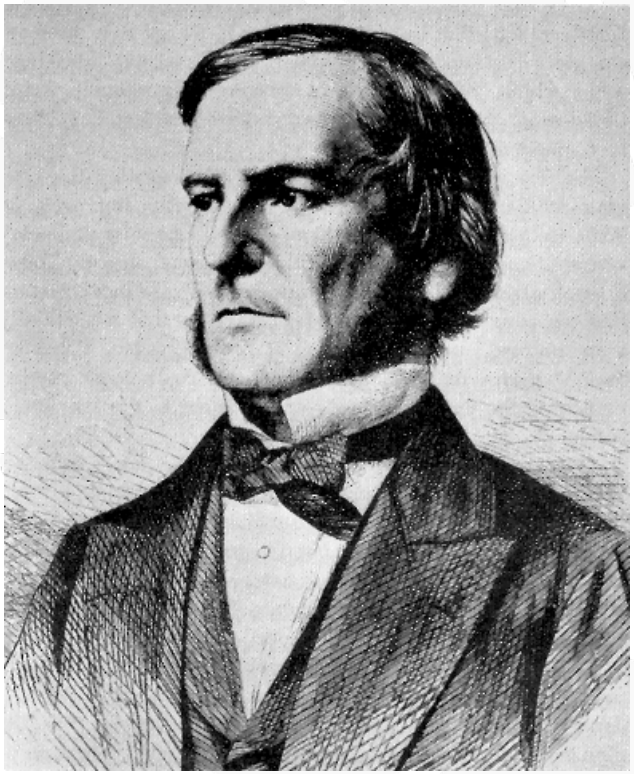
George Boole (02/09/1815 – 08/12/1864)



Principais áreas de atuação:

- Matemática.

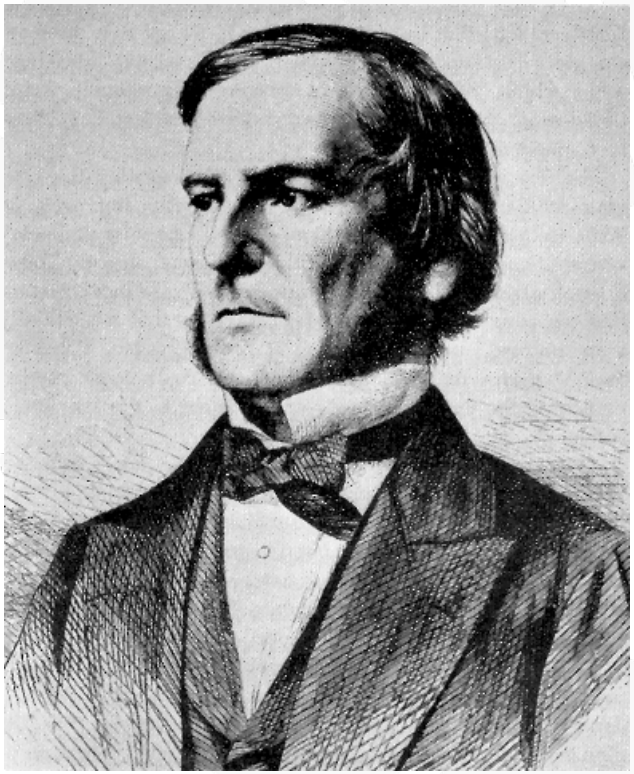
George Boole (02/09/1815 – 08/12/1864)



Principais áreas de atuação:

- Matemática.
- Filosofia.

George Boole (02/09/1815 – 08/12/1864)



Principais áreas de atuação:

- Matemática.
- Filosofia.
- Lógica.

Álgebra booleana

Ramificação de álgebra em que:

Álgebra booleana

Ramificação de álgebra em que:

- As variáveis tomam valores de **verdade**: **Verdadeiro** ou **Falso** (**1** ou **0**).

Álgebra booleana

Ramificação de álgebra em que:

- As variáveis tomam valores de **verdade**: **Verdadeiro** ou **Falso** (**1** ou **0**).
- Os operadores são lógicos, sendo os principais: conjunção (**e**) denotado por $\wedge$, disjunção (**ou**) denotado por $\vee$ e negação (**não**) denotado por $\neg$.

Operações lógicas

Operador	Uso	Significado

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?
$\neg$	$\neg x$	x é falso?

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?
$\neg$	$\neg x$	x é falso?

Tabelas de verdade:

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?
$\neg$	$\neg x$	x é falso?

Tabelas de verdade:

$\wedge$	1	0
1	1	0
0	0	0

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?
$\neg$	$\neg x$	x é falso?

Tabelas de verdade:

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

Operações lógicas

Operador	Uso	Significado
$\wedge$	$x \wedge y$	x e y são verdade?
$\vee$	$x \vee y$	x é verdade ou y é verdade?
$\neg$	$\neg x$	x é falso?

Tabelas de verdade:

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

$\neg$	1	0
	0	1

Expressões lógicas em Python

Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

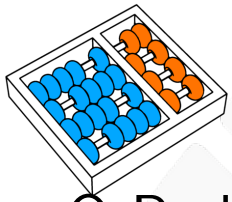
Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a



Operadores lógicos

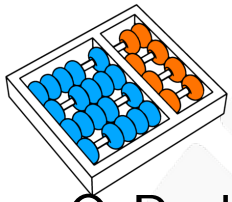
O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False

Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

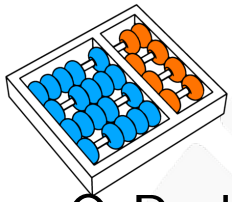
a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False



Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

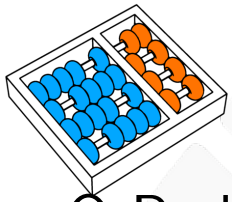
a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True



Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

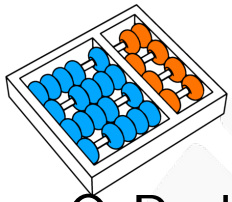


Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:



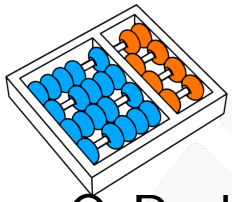
Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:

- **a** e **b** podem ser quaisquer expressões booleanas.



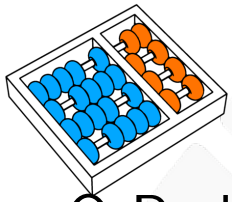
Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:

- **a** e **b** podem ser quaisquer expressões booleanas.
- Podemos escrever expressões do tipo **a and not b or c**.



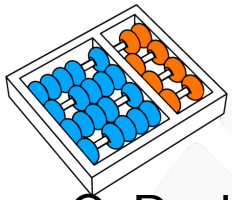
Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:

- **a** e **b** podem ser quaisquer expressões booleanas.
- Podemos escrever expressões do tipo **a and not b or c**.
- **not** precede **and** que precede **or**



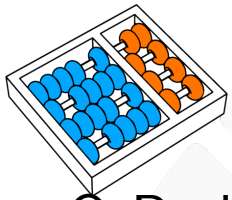
Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:

- **a** e **b** podem ser quaisquer expressões booleanas.
- Podemos escrever expressões do tipo **a and not b or c**.
- **not** precede **and** que precede **or**
- Mais fácil usar parênteses do que lembrar...



Operadores lógicos

O Python tem três operadores lógicos: **and**, **or** e **not**

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Observações:

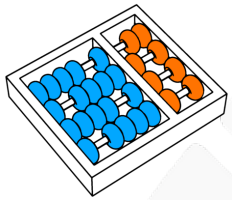
- **a** e **b** podem ser quaisquer expressões booleanas.
- Podemos escrever expressões do tipo **a and not b or c**.
- **not** precede **and** que precede **or**
- Mais fácil usar parênteses do que lembrar...
  - (**a and** (**not b**)) **or c**.

Precedência de operadores

Ordem	Operadores

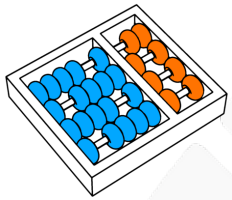
Precedência de operadores

Ordem	Operadores
1º	**



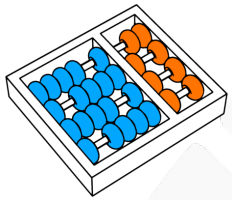
Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //



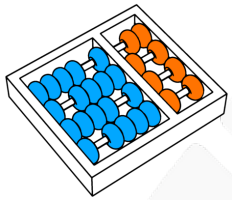
Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -



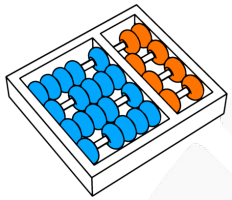
Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -
4º	< > <= >= == !=



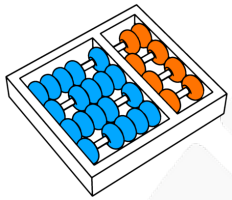
Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -
4º	< > <= >= == !=
5º	not



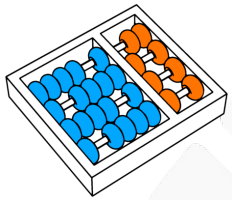
Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -
4º	< > <= >= == !=
5º	not
6º	and



Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -
4º	< > <= >= == !=
5º	not
6º	and
7º	or



Precedência de operadores

Ordem	Operadores
1º	**
2º	* / % //
3º	+ -
4º	< > <= >= == !=
5º	not
6º	and
7º	or

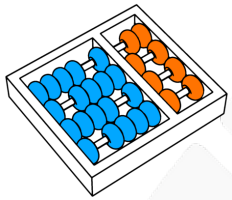
Sugestão: Agrupar sempre com parêntesis (ajuda na leitura)!!

Exemplo

```
1 n = int(input())
2 if n % 2 == 0 or n % 3 == 0:
3     print(n, "é divisível por 2 ou por 3")
4 else:
5     print(n, "não é divisível por 2 e", end=" ")
6     print("não é divisível por 3")
```

Três soluções de um problema

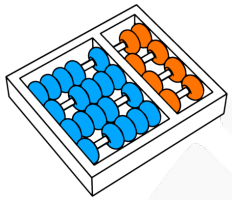
Determinar se n não é divisível por 3.



Três soluções de um problema

Determinar se n não é divisível por 3.

```
1  n = int(input())
2  if n % 3 != 0:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```

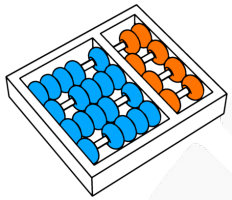


Três soluções de um problema

Determinar se n não é divisível por 3.

```
1  n = int(input())
2  if n % 3 != 0:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```

```
1  n = int(input())
2  if not n % 3 == 0:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```



Três soluções de um problema

Determinar se n não é divisível por 3.

```
1  n = int(input())
2  if n % 3 != 0:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```

```
1  n = int(input())
2  if not n % 3 == 0:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```

```
1  n = int(input())
2  if n % 3 == 1 or n % 3 == 2:
3      print(n, "não é divisível por 3")
4  else:
5      print(n, "é divisível por 3")
```

Regras de De Morgan

Uma regra útil da lógica:

Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.

Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Isso permite:

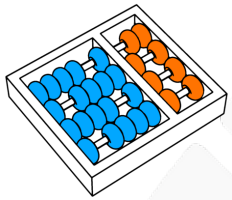
Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Isso permite:

- Escrever expressões menores ou mais legíveis.



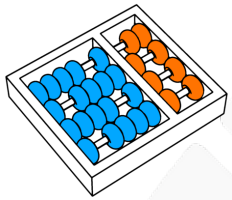
Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Isso permite:

- Escrever expressões menores ou mais legíveis.
- Ou saber porque uma condição falhou.



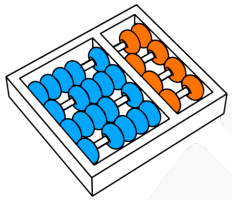
Regras de De Morgan

Uma regra útil da lógica:

- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Isso permite:

- Escrever expressões menores ou mais legíveis.
- Ou saber porque uma condição falhou.
  - Se o **if a and b** falhou é porque



Regras de De Morgan

Uma regra útil da lógica:

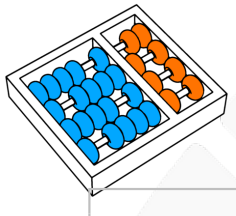
- **not (a or b)** é equivalente a **(not a) and (not b)**.
- **not (a and b)** é equivalente a **(not a) or (not b)**.

Isso permite:

- Escrever expressões menores ou mais legíveis.
- Ou saber porque uma condição falhou.
  - Se o **if a and b** falhou é porque
  - **not a** ou **not b** é verdade.

Voltando a um exercício anterior

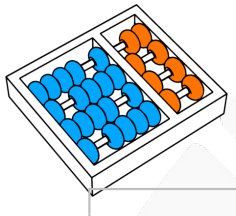
```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```



Voltando a um exercício anterior

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```

Note que um **if** não precisa ter um **else**!

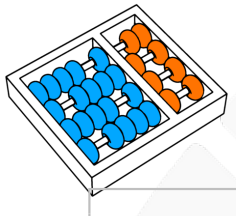


Voltando a um exercício anterior

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```

Note que um **if** não precisa ter um **else**!

- Mas todo **else** precisa ter um **if**...



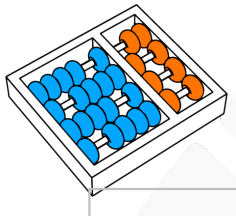
Voltando a um exercício anterior

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```

Note que um **if** não precisa ter um **else**!

- Mas todo **else** precisa ter um **if**...

O que é impresso se **n** for **3**?



Voltando a um exercício anterior

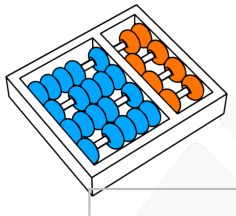
```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```

Note que um **if** não precisa ter um **else**!

- Mas todo **else** precisa ter um **if**...

O que é impresso se **n** for **3**?

- Isso não acontecia no programa anterior...



Voltando a um exercício anterior

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  if not n % 2 == 0:
5      print(n, "não é divisível por 2")
6  if n % 3 == 0:
7      print(n, "é divisível por 3")
```

Note que um **if** não precisa ter um **else**!

- Mas todo **else** precisa ter um **if**...

O que é impresso se **n** for **3**?

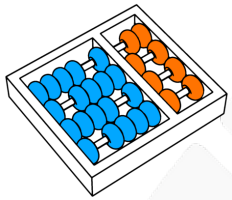
- Isso não acontecia no programa anterior...

Se o primeiro **if** não falhar, ainda assim são verificados os próximos, mas...

- Precisam ser verificados nesse caso?
- Podemos usar De Morgan para ver que, nesse caso, eles vão falhar.
- Como evitar verificações desnecessárias?

Uma versão melhor

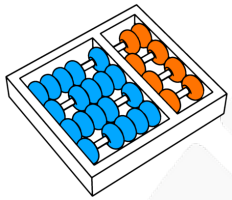
```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  elif n % 2 == 0:
5      print(n, "é divisível por 2 e por 3")
6  elif not n % 3 == 0:
7      print(n, "não é divisível por 2 e por 3")
8  else:
9      print(n, "não é divisível por 2 e é por 3")
10
```



Uma versão melhor

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  elif n % 2 == 0:
5      print(n, "é divisível por 2 e por 3")
6  elif not n % 3 == 0:
7      print(n, "não é divisível por 2 e por 3")
8  else:
9      print(n, "não é divisível por 2 e é por 3")
10
```

O **elif** (de *else if*) testa uma nova condição:

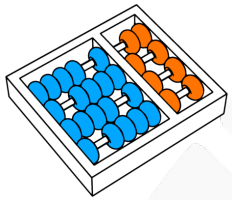


Uma versão melhor

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  elif n % 2 == 0:
5      print(n, "é divisível por 2 e por 3")
6  elif not n % 3 == 0:
7      print(n, "não é divisível por 2 e por 3")
8  else:
9      print(n, "não é divisível por 2 e é por 3")
10
```

O **elif** (de *else if*) testa uma nova condição:

- Apenas se o **if** e os **elif**s anteriores falharam.



Uma versão melhor

```
1  n = int(input())
2  if n % 2 == 0 and (not n % 3 == 0):
3      print(n, "é divisível por 2 e não por 3")
4  elif n % 2 == 0:
5      print(n, "é divisível por 2 e por 3")
6  elif not n % 3 == 0:
7      print(n, "não é divisível por 2 e por 3")
8  else:
9      print(n, "não é divisível por 2 e é por 3")
10
```

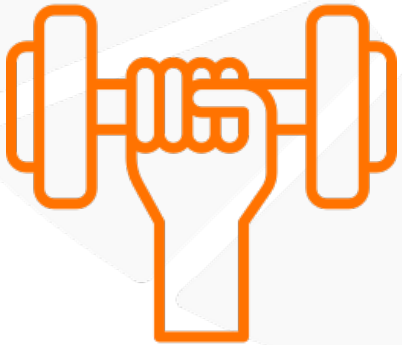
O **elif** (de *else if*) testa uma nova condição:

- Apenas se o **if** e os **elif**s anteriores falharam.

Exercício: Como fazer sem usar **elif**?

Seleção condicional

Vamos fazer alguns exercícios?



Exercício

Vamos fazer um programa em Python que:

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.
- Use a fórmula de Bhaskara.

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.
- Use a fórmula de Bhaskara.

$$\mathbf{x} = \frac{-\mathbf{b} \pm \sqrt{\mathbf{b}^2 - \mathbf{4ac}}}{\mathbf{2a}}$$

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.
- Use a fórmula de Bhaskara.

$$\mathbf{x} = \frac{-\mathbf{b} \pm \sqrt{\mathbf{b}^2 - \mathbf{4ac}}}{\mathbf{2a}}$$

Cuidados:

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.
- Use a fórmula de Bhaskara.

$$\mathbf{x} = \frac{-\mathbf{b} \pm \sqrt{\mathbf{b}^2 - \mathbf{4ac}}}{\mathbf{2a}}$$

Cuidados:

- a não pode ser zero!

Exercício

Vamos fazer um programa em Python que:

- Lê três números a, b e c.
- Calcula as raízes de $ax^2 + bx + c = 0$.
- Use a fórmula de Bhaskara.

$$\mathbf{x} = \frac{-\mathbf{b} \pm \sqrt{\mathbf{b}^2 - \mathbf{4ac}}}{\mathbf{2a}}$$

Cuidados:

- a não pode ser zero!
- $b^2 - 4ac$ não pode ser negativo!

Solução

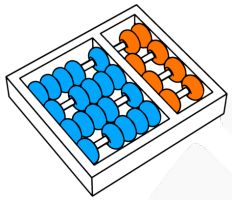
```
1  # equação a x**2 + b x + c = 0
2  a = float(input())
3  b = float(input())
4  c = float(input())
5
6  delta = b**2 - 4 * a * c
7
8  if (a == 0):
9      # Expressão da forma b x + c = 0
10     print("Não é uma equação de segundo grau!")
11 elif (delta < 0):
12     print("As raízes são números complexos!")
13 else:
14     print("Raizes:")
15     print((- b - delta**(1 / 2)) / (2 * a))
16     print((- b + delta**(1 / 2)) / (2 * a))
```

Exercícios

1. Escreva um programa que lê dois números inteiros x e y e diz qual quadrante do espaço (x, y) está.

Exercícios

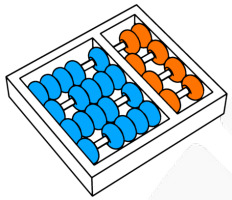
1. Escreva um programa que lê dois números inteiros x e y e diz qual quadrante do espaço (x, y) está.
   - Use operadores lógicos dessa vez...



Exercícios

1. Escreva um programa que lê dois números inteiros x e y e diz qual quadrante do espaço (x, y) está.
   - Use operadores lógicos dessa vez...

2. Escreva um programa que lê três números e encontra o maior dos três.



Exercícios

1. Escreva um programa que lê dois números inteiros x e y e diz qual quadrante do espaço (x, y) está.
   - Use operadores lógicos dessa vez...

2. Escreva um programa que lê três números e encontra o maior dos três.

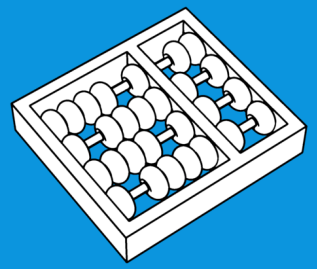
3. Dê um exemplo onde utilizar **if x ... else** é diferente de usar **if x** seguido de **if not x**.

Seleção condicional

MC102 - Algoritmos e Programação de Computadores

03/25

4

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP