



Comparação lógica

Como comparar valores booleanos.



Você entendeu como comparar valores numéricos... E os **valores booleanos**?

```
1 true < false
```

```
2 // ???
```

```
3
```

Os booleanos possuem uma
matemática própria, **precisam dos seus
próprios operadores.**

Operadores booleanos

OU	
E	&&
Não	!

```
1  false || false // false
2  true  || false // true
3  false || true  // true
4  true  || true  // true
5
```

O comparador OU (||) resulta em **true** quando **qualquer valor** for **verdadeiro**.
Caso contrário, resulta em false.

```
1  false && false // false
2  true  && false // false
3  false && true  // false
4  true  && true  // true
5
```


O comparador E (&&) resulta em **true** apenas quando **todos os valores são verdadeiros**.
Caso contrário, resulta em false.

Comparadores lógicos

OU (||)

F	F	F
T	F	T
F	T	T
T	T	T

E (&&)

F	F	F
T	F	F
F	T	F
T	T	T

É possível utilizar **mais de um comparador**
na mesma expressão?

```
1  false || false || true // true
2  false && false && true // false
3  false || true && true // true
4
```

Tabela básica de precedência dos operadores lógicos

E	&&
OU	

```
1 (false || true) && true // true
```

```
2
```

E o operador NOT!?

```
1  !false // true
2  !true  // false
3
4  !(10 < 20) // !true = false
5
```


Tudo que é true **se torna false** e vice versa.

Apesar de se parecerem complexos, os **operadores lógicos são muito especiais na programação.**

```
1  (1 < 2) || (3 > 4) // true || false = true
2  (1 > 2) || (3 > 4) // false || false = false
3
4  (10 < 20) && (20 < 30) // true && true = true
5  (20 < 30) && (30 > 40) // true && false = false
6
7  (20 < 30) && !(30 > 40) // true && !false = true
8
```

```
1  var a = 10
```

```
2  var b = 20
```

```
3  var c = 30
```

```
4
```

```
5  (a < b) && (b < c) // true && true = true
```

```
6
```

Próximos passos

Comparadores lógicos são a **base das estruturas de decisão (IF)**.