BOOLEAN

And BOOLEAN Expressions

by

Lilian Blot

What is a Boolean?

A Boolean ('bool'), is a data type that can take only two values, True or False.

What is a Boolean?

In Python, we can compare numbers using comparison operators such as:

>, <, >=, <=,
== (equal),
!= (not equal).

The result of a comparison expression is a Boolean

Python shell

```
>>> 10 < 7
False
>>> 10 > 7
True
```

Values in variables can be compared, and the result can be stored in a variable too.

Python shell

```
>>> my_condition = 10 < 7
>>> type(my_condition)
<type 'bool'>
>>> small = 6
>>> big = 2000
>>> is_greater = (big >= small)
>>> print(is_greater)
True
```

Pitfalls & Common Mistakes

Make sure to compare comparable objects

Python shell

```
>>> 7 <= '6'
True
>>> 7 >= '6'
False
>>> '7' >= '6'
True
```

Note that strings are compared alphabetically

Pitfalls & Common Mistakes

Exact equality between floating-point numbers is a dangerous concept in Python.

Python shell

```
>>> a = 6458.0/100.0
>>> b = 6200.0/100.0
>>> a-b == 2.58
False
```

To compare two floats, import the `math` module and use the `math.isclose()` method

Python shell

```
>>> import math
>>> math.isclose(a-b, 2.58)
True
```

There are three Boolean operators shown in order of precedence:

1. `not`
2. `and`
3. `Or`

p	not p
True	False
False	True

p	q	p and q
True	True	True
True	False	False
False	True	False
False	False	False

p	q	p or q
True	True	True
True	False	True
False	True	True
False	False	False

Boolean Expressions are evaluated from left to right.

`a and b:`

if **a** is False, b is not evaluated, and the expression returns False.

`a or b:`

if **a** is True, b is not evaluated, and the expression returns True.

Operators Precedence

```
not a  and  not b    and c  or  not d
```

Operators Precedence

```
(not a) and (not b)  and c  or (not d)
```

Operators Precedence

```
((not a) and (not b)) and c  or (not d)
```

Operators Precedence

```
(((not a) and (not b)) and c) or (not d)
```

Operators Precedence

```
(((not a) and (not b)) and c) or (not d))
```

Operators Precedence

```
(((not a) and (not b)) and c) or (not d))
```

Use brackets to make your code more readable and avoid logical errors

Now that we have seen the Boolean type and how to construct Boolean expressions, in our next video we will be looking into a more complex and very useful instruction, the selection statement.