CONDITIONALS

An Introduction to Computer Science

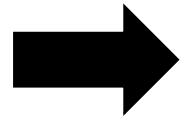
Let's learn about Conditionals.

Purpose

One of the major reasons that programs are useful is because they can do different things depending on their inputs.
To be able to make these decisions, we need to be able to write "conditional expressions". Think of these as a question.

Comparison Operators

```
==
!=
<
<=
>
>=
```



```
True

False
```

Often, we want to know about the relationship between two numbers.
For this, we use the comparison operators.
Each of these operators takes two numbers and returns either True or False.

== (Equal Operator)

```
> 5 == 5
True

> 10 == 7
False
```

You need two equal signs!

Okay, this is gonna sound weird, but in Python we use two equal signs to test for equality. Do not use one equal sign, that means something different.

!= (Not equal operator)

```
> 5 != 5
False

> 10 != 7
True
```

This will be even weirder.
In Python, to test if two things are NOT equal, we use an exclamation mark and an equal sign.

<, <=, >, >= (Greater and Less Than)

```
> 5 < 10
True
> 5 <= 5
True
> 10 > 5
True
> 10 >= 10
True
```

There are four other operators:
The less than operator, the greater than operator, the less than or equal to operator, and the greater than or equal to operator.

Returning True or False

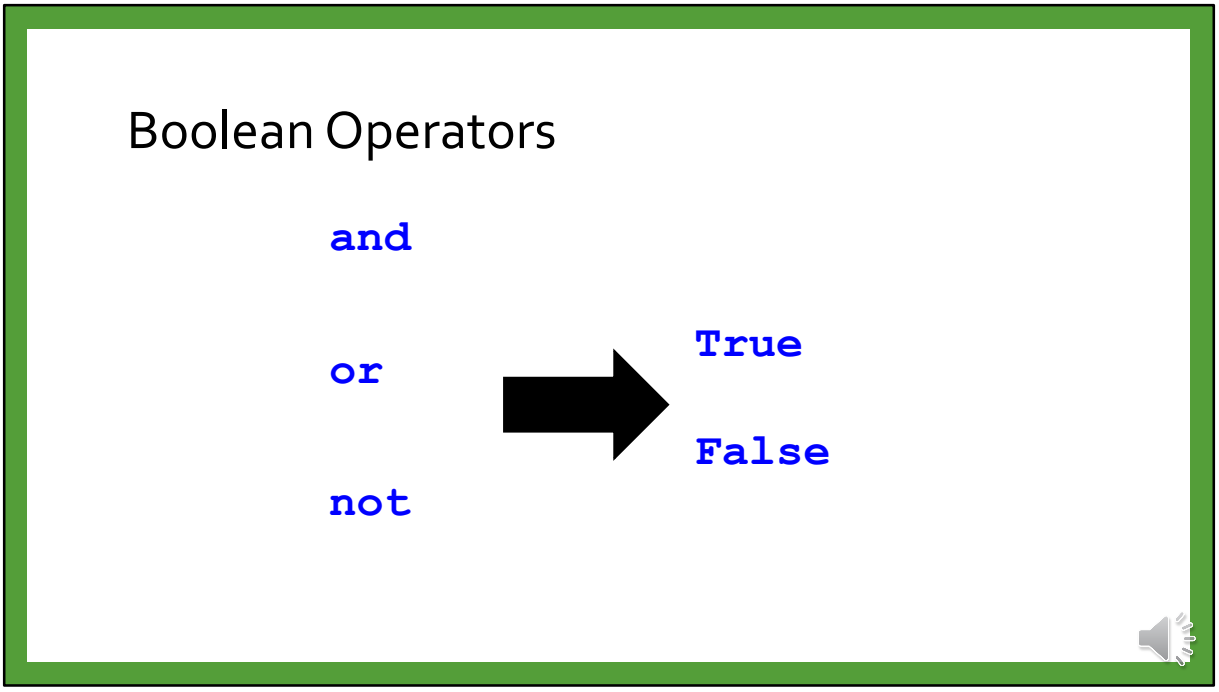
```
> 10 == 10
True
```

The comparison becomes True in the computer's head

```
> print(10 != 10)
False
```

Prints False instead of "10 != 10"

I said before that the operators return either True or False.
Mentally imagine the result of these operations being replaced by True or False.
If you print the value, it will literally be True or False.



In the same way we can add and subtract numeric expressions together, we can also combine booleans expressions together.
There are three operators for this: And, Or, and Not

And

```
> 4<5 and 5<6
True

> 5<4 and 5<6
False

> 5<4 and 6<5
False
```

Left	Right	Result
False	False	False
True	False	False
False	True	False
True	True	True

And returns True if both the left and right expressions are True.

Or

```
> 4<5 or 5<6
True

> 5<4 or 5<6
True

> 5<4 or 6<5
False
```

Left	Right	Result
False	False	False
True	False	True
False	True	True
True	True	True

Or returns True if either the left or right expressions are True.

Not

```
> not 4<5
False

> not 5<4
True
```

Value	Result
False	True
True	False

Not returns True if the expression is False.
Unlike AND and OR, the NOT expression only takes in a single value.

Nesting Conditionals

```
> (5 < 4 and 3 > 7) or (not False and 3 < 2)
False
> not (not (not (not True)))
True
```

You can connect conditionals into fairly complex expressions, just like you could with math.

Conditionals Are Not Distributive

```
5 < 1 or 2
```

Wrong!

```
5 < 1 or 5 < 2
```

Correct!

A common beginner mistake is to think that you can distribute conditionals.
You might think that the code shown here asks if the number 5 is less than 1 or less than 2.
But the "or" operator makes the 2 evaluated separately.
To properly ask this question, you need to write the statement below, with the less than operator used a second time.