

Boolean operators

Introduction to Programming in Python

Rae Harbird, r.harbird@ucl.ac.uk

Boolean Values

George Boole developed a branch of mathematics, Boolean algebra, which was fundamental to the dawn of the Information Age.

Boolean values can be either True or False. So far we've looked at numbers and strings in Python but we can also store boolean values in variables:

```
hate_spiders = True

if hate_spiders:
    print("Spiders are not popular")
```

Logical operators

We can use the **boolean operators**, also called logical operators, and (AND), or (OR) and not (NOT) to generate new boolean values from existing ones.

Truth tables

p	not p
True	False
False	True

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

Examples

Example	Value
<code>(1 == 2) or (8 == 4+4)</code>	True
<code>(1 == 2) and (8 == 4+4)</code>	False
<code>((1<2) or (8 == 4+4)) and (3>4)</code>	False
<code>(1<2) or ((8 == 4+4) and (3>4))</code>	True
<code>not(1<2) or ((8 == 4+4) and (3>4))</code>	False

Guessing two Numbers simplified

Take a look at the program “guess_two_randoms_simplified.py”.

Here is an extract:

```
first_answer = random.randint(MIN, MAX)
second_answer = random.randint(MIN, MAX)

first_guess = int(input("\n\tGuess the first number between "
                        + str(MIN) + " and " + str(MAX) + ": "))
second_guess = int(input("\n\tGuess the second number between "
                        + str(MIN) + " and " + str(MAX) + ": "))

if first_guess == first_answer:
    if second_guess == second_answer:
        print("\tCorrect - well done!")
    else:
        print("\tNo, the answers were ", end='')
        print(str(first_answer) + " and "
              + str(second_answer) + ".\n")
else:
    print("\tNo, the answers were ", end='')
    print(str(first_answer) + " and " + str(second_answer) + ".\n")
```

Guessing in either order

Now take a look at the program `guess_two_randoms_either_order.py`.

```
first_answer = random.randint(MIN, MAX)
second_answer = random.randint(MIN, MAX)

first_guess = int(input("\n\tGuess the first number between "
                        + str(MIN) + " and " + str(MAX) + ": "))
second_guess = int(input("\n\tGuess the second number between "
                        + str(MIN) + " and " + str(MAX) + ": "))

if ((first_guess == first_answer and
    second_guess == second_answer) or

    (first_guess == second_answer and
    second_guess == first_answer)):

    print("\tCorrect - well done!")
else:
    print("\tNo, the answers were ", end='')
    print(str(first_answer) + " and " + str(second_answer) + ".\n")
```


Guessing in Either Order Improved

Improved again in guess_two_randoms_boolean.py.

```
first_answer = random.randint(MIN, MAX)
second_answer = random.randint(MIN, MAX)

first_guess = int(input("\n\tGuess the first number between "
                        + str(MIN) + " and " + str(MAX) + ": "))
second_guess = int(input("\n\tGuess the second number between "
                        + str(MIN) + " and " + str(MAX) + ": "))

okSameOrder = first_guess == first_answer
                and second_guess == second_answer
okOtherOrder = first_guess == second_answer
                and second_guess == first_answer

if okSameOrder or okOtherOrder:
    print("\tCorrect - well done!")
else:
    print("\tNo, the answers were ", end='')
    print(str(first_answer) + " and "
          + str(second_answer) + ".\n")
```

Precedence again

Operator	Description
**	exponentiation
+, -	unary operators
,/,//,%	multiplicative operators
+, -	additive operators
<, <=, >, >=, !=, ==	relational operators
==, !=	equality operators
not x	Boolean NOT
and	Boolean AND
or	Boolean OR
=, +=, -=, *=, /=, %=	assignment operators

Shorthand (1/2)

Readability of your code is important. Python has a feature that makes testing the value of boolean variables more readable.

► Don't do this

```
invalid_input = True
```

```
if invalid_input == True: # No, no, no!!!!  
    print("Try again")
```

► Use this shorthand

```
if invalid_input:  
    print("Try again")  
  
if not invalid_input:  
    print("Well done!")
```

Chaining relational operators

In mathematics, we are familiar with the idea that we can chain relational operators:

$100 \leq x \leq 200$

You can do this in Python too ... but there is some debate over whether you should as a beginner. Most programming languages don't allow you to use this shortcut and you must write:

$100 \leq x \text{ and } x \leq 200$

Test yourself

Expression	Value	Comment
3 <= 4	True	3 is less than 4.
3 =< 4	Error	The "less than or equal" operator is <=, not =<.
3 > 4	False	> is the opposite of <=.
4 > 4	False	
4 <= 4	True	Both sides are equal.
3 == 5 - 2	True	== tests for equality.
3 != 5 - 1	True	!= tests for inequality. It is true that 3 is not 5 - 1.
3 = 6 / 2	Error	Use == to test for equality.
1.0 / 3.0 == 0.333333333	False	A fractional value will not have this level of accuracy.
"10" > 5	Error	You cannot compare a string to a number.