w3 schools

Menu ▾

Log in

Python If ... Else

〈 Previous   Next 〉

Python Conditions and If statements

Python supports the usual logical conditions from mathematics:

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`

These conditions can be used in several ways, most commonly in "if statements" and loops.

An "if statement" is written by using the `if` keyword.

Example

Get your own Python Server

If statement:

```
a = 33
b = 200
```

Dark mode

Try it Yourself »

In this example we use two variables, `a` and `b` , which are used as part of the if statement to test whether `b` is greater than `a` . As `a` is `33` , and `b` is `200` , we know that 200 is greater than 33, and so we print to screen that "b is greater than a".

Indentation

Python relies on indentation (whitespace at the beginning of a line) to define scope in the code. Other programming languages often use curly-brackets for this purpose.

Example

If statement, without indentation (will raise an error):

```python
a = 33
b = 200
if b > a:
print("b is greater than a") # you will get an error
```

Try it Yourself »

Dark mode

◀ ◼ ▶

Elif

The `elif` keyword is Python's way of saying "if the previous conditions were not true, then try this condition".

Example

Get your own Python Server

```python
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

Try it Yourself »

In this example `a` is equal to `b`, so the first condition is not true, but the `elif` condition is true, so we print to screen that "a and b are equal".

Else

The `else` keyword catches anything which isn't caught by the preceding conditions.

Dark mode

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

Try it Yourself »

In this example `a` is greater than `b` , so the first condition is not true, also the `elif` condition is not true, so we go to the `else` condition and print to screen that "a is greater than b".

You can also have an `else` without the `elif` :

Example

Get your own Python Server

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

Try it Yourself »

Short Hand If

If you have only one statement to execute, you can put it on the same line as the if statement.

Dark mode

One line if statement:

```
if a > b: print("a is greater than b")
```

Try it Yourself »

Short Hand If … Else

If you have only one statement to execute, one for if, and one for else, you can put it all on the same line:

Example

Get your own Python Server

One line if else statement:

```
a = 2
b = 330
print("A") if a > b else print("B")
```

Try it Yourself »

This technique is known as **Ternary Operators**, or **Conditional Expressions**.

You can also have multiple else statements on the same line:

Example

Get your own Python Server

One line if else statement, with 3 conditions:

Dark mode

```
print("A") if a > b else print("=") if a == b else print("B")
```

Try it Yourself »

And

The `and` keyword is a logical operator, and is used to combine conditional statements:

Example

Get your own Python Server

Test if `a` is greater than `b`, AND if `c` is greater than `a`:

```
a = 200
b = 33
c = 500
if a > b and c > a:
   print("Both conditions are True")
```

Try it Yourself »

Or

The `or` keyword is a logical operator, and is used to combine conditional statements:

Example

Get your own Python Server

Test if `a` is greater than `b`, OR if `a` is greater than `c`:

```
a = 200
b = 33
```

Dark mode

≡ ⌂ **HTML** **CSS** ◑ 🌐 🔍

```python
print("At least one of the conditions is True")
```

Try it Yourself »

Not

The `not` keyword is a logical operator, and is used to reverse the result of the conditional statement:

Example

Get your own Python Server

Test if `a` is NOT greater than `b` :

```python
a = 33
b = 200
if not a > b:
    print("a is NOT greater than b")
```

Try it Yourself »

Nested If

You can have `if` statements inside `if` statements, this is called *nested* `if` statements.

Example

Get your own Python Server

```python
x = 41

if x > 10:
    print("Above ten,")
    if x > 20:
```

Dark mode

☰   🏠   **HTML**   **CSS**   ◑   🌐   🔍

```
    print("but not above 20.")
```

Try it Yourself »

The pass Statement

`if` statements cannot be empty, but if you for some reason have an `if` statement with no content, put in the `pass` statement to avoid getting an error.

Example   **Get your own Python Server**

```python
a = 33
b = 200

if b > a:
  pass
```

Try it Yourself »

Test Yourself With Exercises

Exercise:

Print "Hello World" if `a` is greater than `b` .

Dark mode

```
a = 50
b = 10
    a    b
  print("Hello World")
```

Submit Answer »

Start the Exercise

❮ Previous   Next ❯

ADVERTISEMENT

Dark mode

2022 Cadillac's On Sale

New Cadillac's Are Turning Heads And Finally On Sale. Discover Deals With Top Searches

All Things Auto

Open >

Dark mode

COLOR PICKER

**Get certified
by completing
a Python
course today!**

w3schools

CERTIFIED · 2023

Get started

Dark mode

HTML   CSS

ADVERTISEMENT

Dark mode

Report Error

Spaces

Upgrade

Newsletter

Get Certified

Top Tutorials

Dark mode

JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Get Certified

HTML Certificate
CSS Certificate
JavaScript Certificate
Front End Certificate
SQL Certificate
Python Certificate
PHP Certificate
jQuery Certificate
Java Certificate
C++ Certificate

Dark mode

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our terms of use, cookie and privacy policy.