Python Operators



# **Python Operators**



21/06/24, 16:54

Next >

## **Python Operators**

Operators are used to perform operations on variables and values.

In the example below, we use the + operator to add together two values:



Python divides the operators in the following groups:

- · Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- · Membership operators
- Bitwise operators

#### **Python Arithmetic Operators**

Arithmetic operators are used with numeric values to perform common mathematical operations:

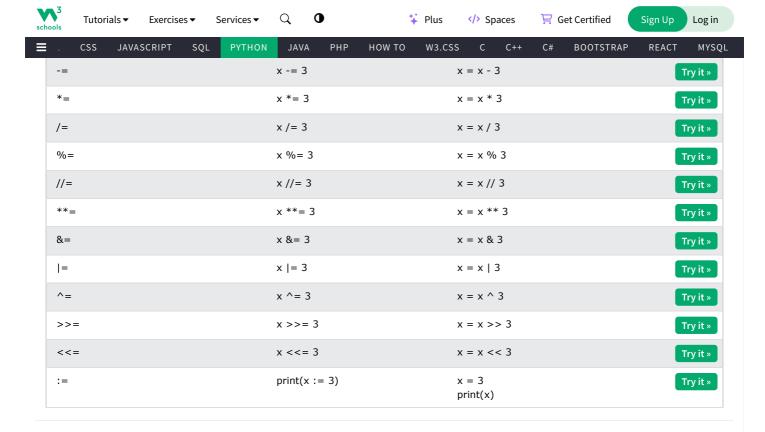
Operator	Name	Example	Try it
+	Addition	x + y	Try it »
-	Subtraction	x - y	Try it »
*	Multiplication	x * y	Try it »
/	Division	x / y	Try it »
%	Modulus	x % y	Try it »
**	Exponentiation	x ** y	Try it »
//	Floor division	x // y	Try it »

# **Python Assignment Operators**

Assignment operators are used to assign values to variables:



Operator	Example	Same As	Try it
----------	---------	---------	--------



#### **Python Comparison Operators**

Comparison operators are used to compare two values:

Operator	Name	Example	Try it
==	Equal	x == y	Try it »
!=	Not equal	x != y	Try it »
>	Greater than	x > y	Try it »
<	Less than	x < y	Try it »
>=	Greater than or equal to	x >= y	Try it »
<=	Less than or equal to	x <= y	Try it »

# **Python Logical Operators**

Logical operators are used to combine conditional statements:

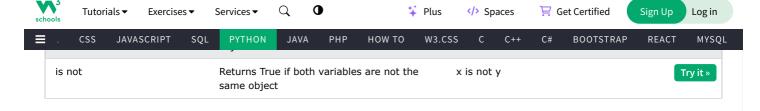
Operator	Description	Example	Try it
and	Returns True if both statements are true	x < 5  and  x < 10	Try it »
or	Returns True if one of the statements is true	x < 5 or x < 4	Try it »
not	Reverse the result, returns False if the result is true	not(x < 5  and  x < 10)	Try it »

### **Python Identity Operators**



Identity operators are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

21/06/24, 16:54 Python Operators



### Python Membership Operators

Membership operators are used to test if a sequence is presented in an object:

Operator	Description	Example	Try it
in	Returns True if a sequence with the specified value is present in the object	x in y	Try it »
not in	Returns True if a sequence with the specified value is not present in the object	x not in y	Try it »

### **Python Bitwise Operators**

Bitwise operators are used to compare (binary) numbers:

Operator	Name	Description	Example	Try it
&	AND	Sets each bit to 1 if both bits are 1	x & y	Try it »
I	OR	Sets each bit to 1 if one of two bits is 1	x   y	Try it »
^	XOR	Sets each bit to 1 if only one of two bits is 1	x ^ y	Try it »
~	NOT	Inverts all the bits	~x	Try it »
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off	x << 2	Try it »
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off	x >> 2	Try it »

### **Operator Precedence**

Operator precedence describes the order in which operations are performed.

#### Example

Parentheses has the highest precedence, meaning that expressions inside parentheses must be evaluated first:

```
print((6 + 3) - (6 + 3))
```

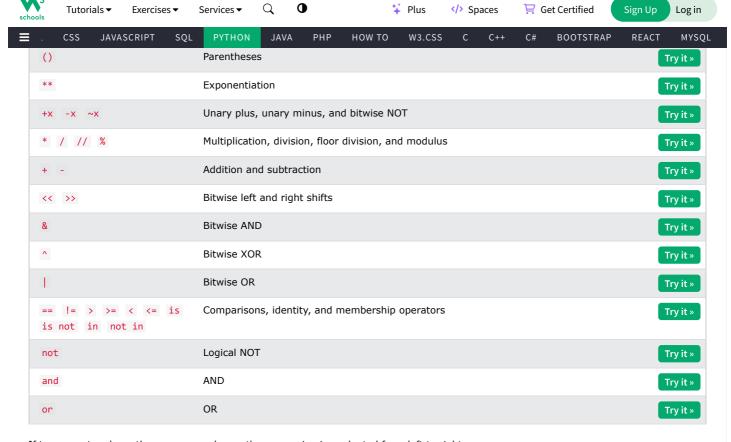
Run example »

#### Example

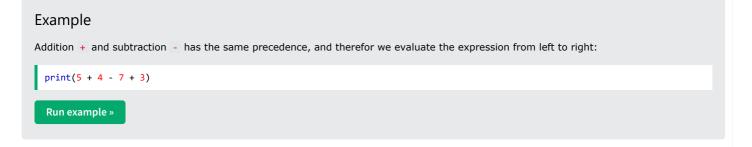
Multiplication \* has higher precedence than addition +, and therefor multiplications are evaluated before additions:

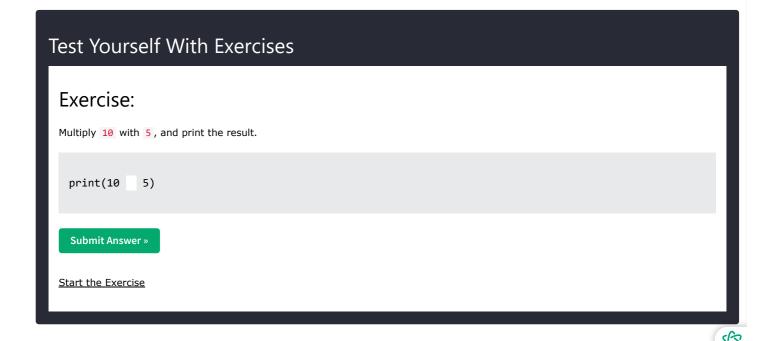
print(100 + 5 \* 3)

Run example »



If two operators have the same precedence, the expression is evaluated from left to right.

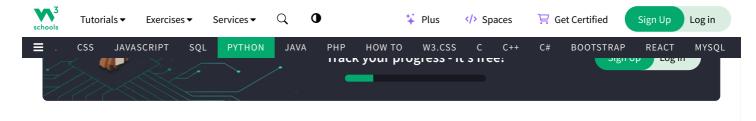




**C** Previous

Next >

21/06/24, 16:54 Python Operators

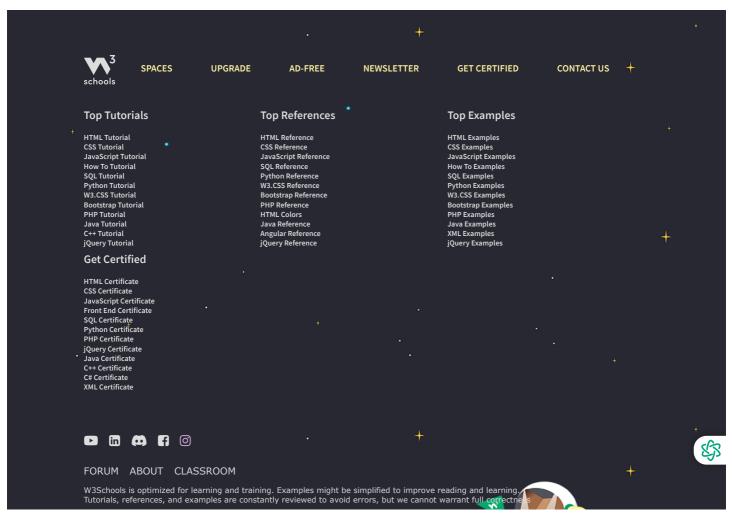




**COLOR PICKER** 







21/06/24, 16:54 Python Operators

