PYTHON.PROGRAMMERS.IM

Python syntax basics

A short official cheat sheet, from the Python Tutorial.

The examples and wording below come from "An Informal Introduction to Python" and "More Control Flow Tools", in the official Python 3.15.2 documentation.[1,2] Type the parts after the >>> prompt if you want to try them in the interpreter.

Comments

Comments start with the hash character, #, and extend to the end of the physical line. A comment may appear at the start of a line or following whitespace or code, but not within a string literal. A hash inside a string is just a hash.[1]

```
# this is a comment
8 / 5  # division always returns
       # a floating-point number
```

Numbers

The interpreter acts as a simple calculator. The operators +, -, * and / perform arithmetic; parentheses group expressions. Division always returns a floating-point number.[1]

```
>>> 2 + 2
4
>>> 50 - 5 * 6
20
>>> 8 / 5
1.6
```

if

The official tutorial calls `if` perhaps the most well-known statement type. It uses `if`, `elif` and `else`, with indented blocks.[2]

```
if x < 0:
    x = 0
    print('Negative changed to zero')
elif x == 0:
    print('Zero')
elif x == 1:
    print('Single')
else:
    print('More')
```

The official example first reads an integer with `int(input(...))`. The branches above are that official example.[2]

Text (strings)

Python can manipulate text, represented by type `str`. Strings can be enclosed in single quotes (`'...'`) or double quotes (`"..."`) with the same result.[1]

```
>>> 'spam eggs'
'spam eggs'
```

Lists

The most versatile compound type is the list: a list of comma-separated values (items) between square brackets. Lists can be indexed and sliced, like other built-in sequences.[1]

```
>>> squares = [1, 4, 9, 16, 25]
>>> squares[0]
1
>>> squares[-1]
25
```

for and range()

Python's `for` statement iterates over the items of any sequence (a list or a string), in the order that they appear in the sequence.[2]

```
words = ['cat', 'window', 'defenestrate']
for w in words:
    print(w, len(w))
```

To iterate over a sequence of numbers, use `range()`. The given end point is never part of the generated sequence; `range(10)` generates 10 values.[2]

```
for i in range(5):
    print(i)
# 0 1 2 3 4
```

Defining a function

The official tutorial introduces `def` with a short Fibonacci function. The first statement can be a documentation string in triple quotes.[2]

```
def fib(n):
    """Print a Fibonacci series less than n."""
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
```

Quick quiz

1. Can a # comment start in the middle of a string literal?
2. Does `8 / 5` return an integer, or a floating-point number?
3. Is the end point of `range(5)` included in the sequence?

Answers. 1. No — a hash inside a string is just a hash.[1] 2. A floating-point number (1.6).[1] 3. No — the end point is never part of the generated sequence; you get 0, 1, 2, 3 and 4.[2]

Sources, checked 17 November 2026, Python 3.15.2 documentation. 1. Tutorial: An Informal Introduction to Python https://docs.python.org/3/tutorial/introduction.html 2. Tutorial: More Control Flow Tools https://docs.python.org/3/tutorial/controlflow.html The `if` sample is the official tutorial example (after the input line).

ReallyByte — Fun. Educational. Useless.™ · python.programmers.im · Free educational fact sheet. Not affiliated with the Python Software Foundation.