Welcome to Python

Video: Why Python?

Reading: Visual Studio Code Reading: Installing Python paths (Optional for Windows Users)

Reading: Installing Python paths (Optional for Mac users)

Reading: Required dependencies

✔ Video: Environment check for

✓ Video: Running code - Command line VS IDE

Video: Python syntax, spaces

2 min

Programming Video: Introduction to Programming Next >

< Previous

Programming in Python > Week 1 > Basic Data type and Function Cheatsheet

Basic Data type and Function Cheatsheet

Here's a quick reference for data types in Python.

Data types

Data type	Meaning	Example
string	Text	'Hello', 'Testing 123'
integer	Numbers	-5, 4, 3, 2, 0
float	Decimals	2.4, 5.2, 1000.00

Flow Control

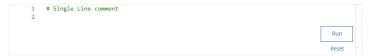
Comparison operators

Operator	Meaning	Example
==	Equals	a == b
!=	Not Equal	a != b
<	Less than	a < b
>	Greater than	a > b
<=	Less than or Equal to	a <= b
>=	Greater than or Equal to	a >= b

Comments

Single-line comments

Placing a # symbol in front of the text you want to be a comment causes Python to ignore everything from that point until the end of the current line.



Multi-line comments

Python does not really have a method for multi line comments, so a # symbol can be used on every line.

```
# This is a multiline comment
# which can be used for long comments
                                                                                                                                          Run
```

Inline/code comments

The # symbol will cause Python to ignore everything from that point until the end of the current line, so inline comments can be created in this way.

```
x = 1 \text{ # assigns value of 1 to } x
```

Built-in Functions

print()

This function looks for the default output device, your terminal, and displays the value passed to it.

```
print("Hello")
```

input()

 $This function \ looks for the \ default input \ device, your \ keyboard, and \ captures \ the \ value. \ This \ value \ can \ then \ be \ assigned$

```
print("Where do you live?")
location = input()
print("So you live in " + location)
                                                                                                                                                                              Run
```

This function returns the length or the count of the elements contained within the structure it is applied on. This may be a string, array, list, tuple, dictionary or any sequence.

```
1 len("Hello")
2 5
                                                                                        Run
```

str()

This function can be used to convert the provided value into a ${\tt String}$

int()

This function can be used to convert the provided value into an ${\tt int}$

```
1 int('75')
2 75
                                                                                         Run
```

float()

This function can be used to convert the provided value into a **float**

```
some_int = 10
float(some_int)
10.0
                                                                                                                     Run
```

Creating Functions

Functions in Python require a keyword to define them: def followed by an identifier (a name) this forms the function signature. The body of the function contains the code to run when the function is called.

```
def say_hello():
    return "Hello there!"
# With parameters
def say_hello(you):
    return "Hello " + you
                                                                                                                                                           Run
```

Mark as completed





🖒 Like 📿 Dislike 🏳 Report an issue