
Python Tutorial

Release 3.7.0

**Guido van Rossum
and the Python development team**

September 02, 2018

**Python Software Foundation
Email: docs@python.org**

CONTENTS

1	Whetting Your Appetite	3
2	Using the Python Interpreter	5
2.1	Invoking the Interpreter	5
2.2	The Interpreter and Its Environment	6
3	An Informal Introduction to Python	9
3.1	Using Python as a Calculator	9
3.2	First Steps Towards Programming	16
4	More Control Flow Tools	19
4.1	if Statements	19
4.2	for Statements	19
4.3	The range() Function	20
4.4	break and continue Statements, and else Clauses on Loops	21
4.5	pass Statements	22
4.6	Defining Functions	22
4.7	More on Defining Functions	24
4.8	Intermezzo: Coding Style	29
5	Data Structures	31
5.1	More on Lists	31
5.2	The del statement	35
5.3	Tuples and Sequences	36
5.4	Sets	37
5.5	Dictionaries	38
5.6	Looping Techniques	39
5.7	More on Conditions	40
5.8	Comparing Sequences and Other Types	40
6	Modules	43
6.1	More on Modules	44
6.2	Standard Modules	46
6.3	The dir() Function	47
6.4	Packages	48
7	Input and Output	53
7.1	Fancier Output Formatting	53
7.2	Reading and Writing Files	57
8	Errors and Exceptions	61

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see [library-index](#). [reference-index](#) gives a more formal definition of the language. To write extensions in C or C++, read [extending-index](#) and [c-api-index](#). There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in [library-index](#).

The [Glossary](#) is also worth going through.